

FILEID**DISKQUOTA

19

```
0001 0
0002 0 MODULE DISKQUOTA {
0003 0   LANGUAGE (BLISS32).
0004 0   MAIN = DISK QUOTA,
0005 0   ADDRESSING_MODE (EXTERNAL = GENERAL,
0006 0           NONEXTERNAL = LONG_RELATIVE),
0007 0   IDENT = 'V04-000'
0008 0   )
0009 1 BEGIN
0010 1
0011 1
0012 1 ****
0013 1 *
0014 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0015 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0016 1 * ALL RIGHTS RESERVED.
0017 1 *
0018 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0019 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0020 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0021 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0022 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0023 1 * TRANSFERRED.
0024 1 *
0025 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0026 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0027 1 * CORPORATION.
0028 1 *
0029 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0030 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0031 1 *
0032 1 *
0033 1 ****
0034 1
0035 1 ++
0036 1
0037 1 FACILITY: VMS System Manager Utilities
0038 1
0039 1 ABSTRACT:
0040 1
0041 1 This program implements the commands necessary to maintain the
0042 1 quota file on a files-11 structure level 2 disk. Functions are
0043 1 provided to create the quota file, enable and disable quotas,
0044 1 add, list, modify, and remove authorization entries.
0045 1
0046 1 ENVIRONMENT:
0047 1
0048 1 VAX/VMS Operating System
0049 1
0050 1 --
0051 1
0052 1
0053 1 AUTHOR: Andrew C. Goldstein, CREATION DATE: 19-Jun-1979 18:54
0054 1
0055 1 MODIFIED BY:
0056 1
0057 1 V03-003 LMP0140      L. Mark Pilant,      23-Aug-1983 12:57
```

58 0058 1 : Add support for alphanumeric UICs.
59 0059 1 :
60 0060 1 : V03-002 LMP0133 L. Mark Pilant, 4-Aug-1983 12:14
61 0061 1 : Don't set the protection for the created quota file. Let
62 0062 1 : the ACP determine it.
63 0063 1 :
64 0064 1 : V03-001 ACG0288 Andrew C. Goldstein, 16-Apr-1982 9:38
65 0065 1 : Add DO_IO entry point for REBUILD
66 0066 1 :
67 0067 1 : V02-006 MLJ0058 Martin L. Jack, 4-Nov-1981 20:16
68 0068 1 : Extend PLIT in ACT_CREATE so that newly initialized quota file
69 0069 1 : does not contain garbage in last few longwords.
70 0070 1 :
71 0071 1 : V02-005 STJ0055 Steven T. Jeffreys, 29-Jun-1981
72 0072 1 : Changed external references to use general addressing mode.
73 0073 1 :
74 0074 1 : V0004 ACG0129 Andrew C. Goldstein, 25-Jan-1980 19:28
75 0075 1 : Use common REBUILD routine
76 0076 1 :
77 0077 1 : V0003 ACG0087 Andrew C. Goldstein,
78 0078 1 : Steve Jeffreys, 20-Nov-1979 20:41
79 0079 1 : Add help facility, remove EXAMINE command, add EXIT command
80 0080 1 : Add overdraft limit, default values for ADD
81 0081 1 :
82 0082 1 : V0002 ACG0056 Andrew C. Goldstein, 8-Aug-1979 14:49
83 0083 1 : Fix REBUILD function to work on non-volume sets
84 0084 1 :
85 0085 1 : **
86 0086 1 :
87 0087 1 :
88 0088 1 : LIBRARY 'SYSSLIBRARY:LIB.L32';
89 0089 1 : LIBRARY 'SYSSLIBRARY:TPAMAC.L32';
90 0090 1 :
91 0091 1 :
92 0092 1 : FORWARD ROUTINE
93 0093 1 : DISK_QUOTA,
94 0094 1 : INV_COMMAND,
95 0095 1 : INV_SWITCH,
96 0096 1 : SAVE_KEY,
97 0097 1 : USE_DEFAULT : NOVALUE,
98 0098 1 : DEF_HANDLER : NOVALUE,
99 0099 1 : ACT_USE,
100 0100 1 : ACT_CREATE,
101 0101 1 : ACT_ENABLE,
102 0102 1 : ACT_DISABLE,
103 0103 1 : ACT_ADD,
104 0104 1 : ACT_REMOVE,
105 0105 1 : ACT_SHOW,
106 0106 1 : ACT MODIFY,
107 0107 1 : ACT_REBUILD,
108 0108 1 : ACT_HELP,
109 0109 1 : MAIN_HANDLER,
110 0110 1 : EXIT_HANDLER : NOVALUE,
111 0111 1 : COMMON_IO;
112 0112 1 :
113 0113 1 : Structure declarations used for system defined structures to
114 0114 1 : save typing.

```
115      0115 1 ! STRUCTURE
116      0116 1     BBLOCK [O, P, S, E; N] =
117      0117 1       [N]
118      0118 1       ((BBLOCK+0)<P,S,E>,
119      0119 1
120      0120 1
121      0121 1     BBLOCKVECTOR [I, O, P, S, E; N, BS] =
122      0122 1       [N=BS]
123      0123 1       ((BBLOCKVECTOR+I*BS)+0)<P,S,E>,
124      0124 1
125      0125 1     EXIT_CTRL_BLK [I : N] =
126      0126 1       [(4+N)*4]                                ! exit handler descriptor
127      0127 1       (EXIT_CTRL_BLK+I*4)<0,32,0>;          ! N = # of arguments ( N <= 1 )
128      0128 1                                         ! the block is a longword array
129      0129 1
130      0130 1 ! Macro to generate a string descriptor.
131      0131 1
132      0132 1 ! MACRO
133      M 0133 1     DESCRIPTOR (STRING) =
134      0134 1       UPLIT (%CHARCOUNT (STRING), UPLIT BYTE (STRING))%
135      0135 1
136      0136 1 ! Macro to signal error exit.
137      0137 1
138      0138 1 ! MACRO
139      M 0139 1     ERR_EXIT [] =
140      M 0140 1       SIGNAL_STOP (%REMAINING)
141      0141 1       %;
142      0142 1
143      0143 1 ! Macro to signal error message.
144      0144 1
145      0145 1 ! MACRO
146      M 0146 1     ERR_MESSAGE [] =
147      M 0147 1       SIGNAL (%REMAINING)
148      0148 1       %;
149      0149 1
150      0150 1 ! Macro to declare argument list in TPARSE action routine.
151      0151 1
152      0152 1 ! MACRO
153      M 0153 1     TPARSE_ARGS =
154      M 0154 1       BUILTIN AP;
155      M 0155 1       BIND TPARSE_BLOCK = AP : REF BBLOCK;
156      0156 1       %;
```

```
158      0157 1 1+
159      0158 1 Error messages
160      0159 1
161      0160 1 Macro to generate each error message.
162      0161 1
163      0162 1
164      0163 1 1-
165      0164 1
166      0165 1 MACRO
167      M 0166 1     ERR_TEXT (CODE, COUNT, SEVERITY, STRING) =
168      M 0167 1         LITERAL %NAME ('DSKQS_',CODE) = MSG_CODE + FAC_CODE*16;
169      M 0168 1         SWITCHES UNAMES;
170      M 0169 1         PSELECT OWN = $MSG_TEXT;
171      M 0170 1         OWN MSG_TEXT : VECTOR [%CHARCOUNT(CODE)+11+%CHARCOUNT(STRING)+2, BYTE]
172      M 0171 1             INITIAL (BYTE (COUNT,
173      M 0172 1                 %CHARCOUNT(CODE)+11+%CHARCOUNT(STRING),
174      M 0173 1                 '$DISKQ-', %STRING (SEVERITY) '-',
175      M 0174 1                 %STRING (CODE), ', ', STRING));
176      M 0175 1         PSELECT OWN = $MSG_INDEX;
177      M 0176 1         OWN MSG_INDEX : INITIAL (MSG_TEXT);
178      M 0177 1         UNDECLARE MSG_TEXT, MSG_INDEX;
179      M 0178 1         SWITCHES NOUNAMES;
180      M 0179 1         %ASSIGN (MSG_CODE, MSG_CODE+8)
181      M 0180 1         PSELECT OWN = $OWNS;
182      0181 1         %;
183      0182 1
184      0183 1 1 Initialize and label the message sections.
185      0184 1
186      0185 1
187      0186 1
188      0187 1 PSELECT
189      0188 1     OWN      = $MSG_TEXT (NOWRITE, ALIGN(0));
190      0189 1     OWN      = MESSAGE_TEXT : VECTOR [0, BYTE];
191      0190 1 PSELECT
192      0191 1     OWN      = $MSG_INDEX (NOWRITE, ALIGN (2));
193      0192 1     OWN      = MESSAGE_TABLE : VECTOR [0];
194      0193 1
195      0194 1
196      0195 1
197      0196 1 COMPILETIME
198      0197 1     MSG_CODE      = 0;
199      0198 1
200      0200 1 1 Generate the error messages
201      0201 1
202      0202 1
203      0203 1 LITERAL
204      0204 1     FAC_CODE      = 69;          ! or whatever
205      0205 1
206      0206 1
207      0207 1     ERR_TEXT      (CMD_ERR,      0, F, 'I/O error reading commands');
208      0208 1     ERR_TEXT      (INV_CMD,      6, E, 'unrecognized command!/!AD\!AD\!AD');
209      0209 1     ERR_TEXT      (AMB_CMD,      6, E, 'ambiguous command!/!AD\!AD\!AD');
210      0210 1     ERR_TEXT      (INV_QUAL,    6, E, 'unrecognized qualifier!/!AD\!AD\!AD');
211      0211 1     ERR_TEXT      (AMB_QUAL,    6, E, 'ambiguous qualifier!/!AD\!AD\!AD');
212      0212 1     ERR_TEXT      (INV_UIC,     6, E, 'invalid UIC!/!AD\!AD\!AD');
213      0213 1     ERR_TEXT      (SYNTAX,      6, E, 'command syntax error!/!AD\!AD\!AD');
```

215	0214	1	ERR_TEXT	(NONLOCAL,	0. E. 'device is not a local device');
216	0215	1	ERR_TEXT	(NOTRAN,	0. E. 'logical name is recursively defined');
217	0216	1	ERR_TEXT	(NODEVICE,	0. E. 'no device currently selected');
218	0217	1	ERR_TEXT	(CREATERR,	0. E. 'error creating quota file');
219	0218	1	ERR_TEXT	(INITERR,	0. E. 'error initializing quota file');
220	0219	1	ERR_TEXT	(CLOSERR,	0. E. 'error closing quota file');
221	0220	1	ERR_TEXT	(ACTERR,	0. E. 'failed to enable quota file');
222	0221	1	ERR_TEXT	(DACTERR,	0. E. 'failed to disable quota file');
223	0222	1	ERR_TEXT	(ADDERR,	0. E. 'failed to add quota file entry');
224	0223	1	ERR_TEXT	(REMOVEERR,	0. E. 'failed to remove quota file entry');
225	0224	1	ERR_TEXT	(MODIFYERR,	0. E. 'failed to modify quota file entry');
226	0225	1	ERR_TEXT	(EXAMINERR,	0. E. 'cannot examine quota file entry');
227	0226	1	ERR_TEXT	(INUSE,	3. I. '[!OW,!OW] has !UL blocks in use');
228	0227	1	ERR_TEXT	(LOCKERR,	0. E. 'failed to lock volume');
229	0228	1	ERR_TEXT	(UNLOCKERR,	0. E. 'failed to unlock volume');
230	0229	1	ERR_TEXT	(MAXVOLS,	0. E. 'volume set has too many volumes to handle');
231	0230	1	ERR_TEXT	(ACCINDEXF,	1. E. 'failed to access index file on relative volume !UW');
232	0231	1	ERR_TEXT	(ACCQFILE,	0. E. 'failed to access quota file');
233	0232	1	ERR_TEXT	(QUOTARERR,	0. E. 'I/O error reading quota file');
234	0233	1	ERR_TEXT	(BITMAPERR,	1. E. 'I/O error reading index file bitmap on relative volume !UW');
235	0234	1	ERR_TEXT	(HEADERERR,	2. W. 'I/O error reading file header !UL on relative volume !UW');
236	0235	1	ERR_TEXT	(MEMALLOC,	0. E. 'cannot allocate sufficient memory');
237	0236	1	ERR_TEXT	(HOMEBLOCK,	1. E. 'failed to read home block on relative volume !UW');
238	0237	1	ERR_TEXT	(HELP_INIT,	1. E. 'failed help library index init');
239	0238	1	ERR_TEXT	(HELP_OPEN,	1. E. 'failed to open help library');
240	0239	1	ERR_TEXT	(HELP_TEXT,	1. E. 'failed to access help text');

242 0240 1 |
243 0241 1 | Module own storage.
244 0242 1 |
245 0243 1 | LITERAL
246 0244 1 | COMMAND_LENGTH = 132,
247 0245 1 | OUTPUT_LENGTH = 132,
248 0246 1 | MAX_KEYS = 14, ! 2*(max # of keys) for HELP command
249 0247 1 |
250 0248 1 | The following are indexes into the Exit Handler Control Block
251 0249 1 |
252 0250 1 | XHNDLR_ADDRESS = 1, ! exit handler address
253 0251 1 | XHNDLR_ARGCNT = 2, ! exit handler argument count
254 0252 1 | XHNDLR_STSADDR = 3; ! system exit status address
255 0253 1 |
256 0254 1 | OWN
257 0255 1 | CHANNEL : WORD ! channel for disk I/O
258 0256 1 | IO_STATUS : VECTOR [4, WORD], ! I/O status block
259 0257 1 | COMMAND_LINE : VECTOR [COMMAND_LENGTH, BYTE], ! command line buffer
260 0258 1 | OUTPUT_LINE : VECTOR [OUTPUT_LENGTH, BYTE], ! output line buffer
261 0259 1 |
262 0260 1 | COMMAND_DESC : VECTOR [2] INITIAL (COMMAND_LENGTH, COMMAND_LINE),
263 0261 1 | ! command line descriptor
264 0262 1 | OUTPUT_DESC : VECTOR [2] INITIAL (OUTPUT_LENGTH, OUTPUT_LINE),
265 0263 1 | ! output line descriptor
266 0264 1 | EXIT_HNDLR_DESC : EXIT_CTRL_BLK [1],
267 0265 1 | ! exit handler descriptor
268 0266 1 |
269 0267 1 |
270 0268 1 | Area to zero before each command.
271 0269 1 |
272 0270 1 | ZERO_AREA : VECTOR [0].
273 0271 1 |
274 0272 1 | Cleanup action flags
275 0273 1 |
276 0274 1 | CLEANUP_FLAGS : BITVECTOR [32];
277 0275 1 |
278 0276 1 | LITERAL
279 0277 1 | CLF_UNLOCK = 0, ! unlock volume set
280 0278 1 | CLF_EXIT = 1; ! exit command entered
281 0279 1 |
282 0280 1 | Quota file record buffers
283 0281 1 |
284 0282 1 | OWN
285 0283 1 | SRC_REC : BBLOCK [DQFSC_LENGTH],
286 0284 1 | DST_REC : BBLOCK [DQFSC_LENGTH];
287 0285 1 |
288 0286 1 | FIB for quota file operations
289 0287 1 |
290 0288 1 | QUOTA_FIB : BBLOCK [FIBSC_LENGTH],
291 0289 1 |
292 0290 1 | TPARSE action routine output
293 0291 1 |
294 0292 1 | UIC_FLAGS : BITVECTOR [32], ! UIC wild card flags
295 0293 1 |
296 0294 1 |
297 0295 1 | Storage used for HELP function.
298 0296 1 |

```
299      0297 1      KEY_VECTOR : VECTOR [MAX_KEYS], ! use as a descriptor vector
300      0298 1      KEY_INDEX,
301      0299 1
302      0300 1      ZERO_END : VECTOR [0];
303      0301 1
304      0302 1      LITERAL ZERO_LENGTH = ZERO_END - ZERO_AREA;
305      0303 1
306      0304 1
307      0305 1      ! Quota record descriptors
308      0306 1
309      0307 1      OWN
310      0308 1      SRCREC_DESC : VECTOR [2] INITIAL (DQFSC_LENGTH, SRC_REC),
311      0309 1      DSTREC_DESC : VECTOR [2] INITIAL (DQFSC_LENGTH, DST_REC),
312      0310 1      QFIB_DESC : VECTOR [2] INITIAL (FIBSC_LENGTH, QUOTA_FIB);
313      0311 1
314      0312 1      ! TPARSE interface and output
315      0313 1
316      0314 1      LITERAL
317      0315 1      WILD_GROUP = $BITPOSITION (FIB$V_ALL_GRP),
318      0316 1      WILD_MEMBER = $BITPOSITION (FIB$V_ALL_MEM),
319      0317 1      PERM_SPEC = $BITPOSITION (FIB$V_MOD_PERMS),
320      0318 1      OVER_SPEC = $BITPOSITION (FIB$V_MOD_OVER);
321      0319 1
322      0320 1      OWN
323      0321 1      TPARSE_BLOCK : BBLOCK [TPASK_LENGTH];
324      0322 1      INITIAL (TPASK_COUNTO, TPASM_ABBREV);
325      0323 1
326      0324 1      BIND
327      0325 1      UIC_VALUE = SRC_REC[DQFSL_UIC], ! full UIC
328      0326 1      PERM_VALUE = SRC_REC[DQFSL_PERMQUOTA], ! permanent quota
329      0327 1      OVER_VALUE = SRC_REC[DQFSL_OVERDRAFT]; ! overdraft limit
330      0328 1
331      0329 1      PSECT PLIT = $OWNS;
332      0330 1
333      0331 1      BIND
334      0332 1      QFILE_NAME = DESCRIPTOR ('QUOTA.SYS;1'); ! quota file name
335      0333 1
336      0334 1      PSECT PLIT = $SPLIT$;
```

```
338 0335 1 GLOBAL ROUTINE DISK_QUOTA =  
339 0336 1  
340 0337 1 ++  
341 0338 1 Functional Description:  
342 0339 1 This is the main program of the disk quota utility. It accepts  
343 0340 1 commands from SYSSINPUT, parses and processes them, and reports  
344 0341 1 errors.  
345 0342 1 Calling Sequence:  
346 0343 1 standard  
347 0344 1 Input Parameters:  
348 0345 1 none  
349 0346 1 Implicit Inputs:  
350 0347 1 none  
351 0348 1 Output Parameters:  
352 0349 1 none  
353 0350 1 Implicit Outputs:  
354 0351 1 none  
355 0352 1 Routines Called:  
356 0353 1 none  
357 0354 1 Routine Value:  
358 0355 1 none  
359 0356 1 Signals:  
360 0357 1 none  
361 0358 1 Side Effects:  
362 0359 1 none  
363 0360 1 --  
364 0361 1 BEGIN  
365 0362 1 LOCAL  
366 0363 2 STATUS, ! general status value  
367 0364 2 P; ! general string pointer  
368 0365 2 : Generate translation table to convert lower case to upper case.  
369 0366 2  
370 0367 2  
371 0368 2  
372 0369 2  
373 0370 2  
374 0371 2  
375 0372 2  
376 0373 2  
377 0374 2  
378 0375 2  
379 0376 2  
380 0377 2  
381 0378 2  
382 0379 2  
383 0380 2  
384 0381 2  
385 0382 2  
386 0383 2 MACRO  
387 M 0384 2 UPCASE_ENTRY (DUMMY) [] =  
388 M 0385 2 %IF ((%COUNT AND XX'7F'), GEQU 'a') AND ((%COUNT AND XX'7F') LEQU 'z')  
389 M 0386 2 %THEN (%COUNT AND XX'5F')  
390 M 0387 2 %ELSE (%COUNT AND XX'7F')  
391 M 0388 2 %FI  
392 M 0389 2 %IF %COUNT LSSU 255  
393 M 0390 2 %THEN , UPCASE_ENTRY (0)  
394 M 0391 2 %FI
```

: 395 0392 2 %:
: 396 0393 2
: 397 0394 2
: 398 0395 2 BIND UPCase_TABLE = UPLIT BYTE (UPCase_ENTRY (0));
: 399 0396 2
: 400 0397 2 EXTERNAL LITERAL LIBS_SYNTAXERR; ! syntax error status from TPARSE
: 401 0398 2
: 402 0399 2 EXTERNAL ROUTINE LIB\$GET_INPUT : ADDRESSING_MODE (GENERAL); ! get line from SYSSINPUT
: 403 0400 2 LIB\$TPARSE : ADDRESSING_MODE (GENERAL); ! parse and process command
: 404 0401 2
: 405 0402 2
: 406 0403 2
: 407 0404 2

```

409      0405 2 | TPARSE state table to parse commands.
410      0406 22 |
411      0407 22 |
412      0408 22 |
413      0409 22 | $INIT_STATE (STATE_TABLE, KEY_TABLE);
414      0410 22 |
415      0411 22 |
416      0412 22 | Initial state - acquire command.
417      0413 22 |
418      0414 22 |
419      P 0415 22 | $STATE (START,
420      P 0416 22     ('ADD',
421      P 0417 22       DO_ADD),
422      P 0418 22     ('CREATE',
423      P 0419 22       MORE, ACT_CREATE),
424      P 0420 22     ('DISABLE',
425      P 0421 22       MORE, ACT_DISABLE),
426      P 0422 22     ('ENABLE',
427      P 0423 22       MORE, ACT_ENABLE),
428      P 0424 22     ('EXIT',
429      P 0425 22       TPAS_EXIT,,T*CLF_EXIT,CLEANUP_FLAGS),
430      P 0426 22     ('HELP',
431      P 0427 22       DO_HELP),
432      P 0428 22     ('MODIFY',
433      P 0429 22       DO_MODIFY),
434      P 0430 22     ('REBUILD',
435      P 0431 22       MORE, ACT_REBUILD),
436      P 0432 22     ('REMOVE',
437      P 0433 22       DO_REMOVE),
438      P 0434 22     ('SHOW',
439      P 0435 22       DO_SHOW),
440      P 0436 22     ('USE',
441      P 0437 22       DO_USE),
442      P 0438 22     (TPAS_SYMBOL.., INV_COMMAND),
443      P 0439 22     (TPAS_EOS,   TPAS_EXIT),
444      P 0440 22     );
445      P 0441 22     |
446      P 0442 22     USE command
447      P 0443 22     |
448      P 0444 22     ADD command
449      P 0445 22     |
450      P 0446 22     |
451      P 0447 22     |
452      P 0448 22     $STATE (DO_ADD,
453      P 0449 22       (((CMD_SWIT), DO_ADD),
454      P 0450 22       ((UICL), DO_ADD1),
455      P 0451 22       );
456      P 0452 22     |
457      P 0453 22     |
458      P 0454 22     MODIFY command
459      P 0455 22     |
460      P 0456 22     |
461      P 0457 22     $STATE (DO_MODIFY,
462      P 0458 22       (((CMD_SWIT), DO_MODIFY),
463      P 0459 22       ((UICL), DO_MODIFY1),
464      P 0460 22       );
465      P 0461 22     );

```

```

466 P 0462 2 $STATE (DO_MODIFY1,
467 P 0463 2 ((CMD_SWIT), DO_MODIFY1),
468 P 0464 2 (TPAS_LAMBDA, MORE, ACT_MODIFY)
469 0465 2 );
470 0466 2
471 0467 2 | SHOW command
472 0468 2
473 0469 2
474 0470 2 | $STATE (DO_SHOW,
475 P 0471 2 ((UIC), MORE, ACT_SHOW)
476 P 0472 2 );
477 0473 2
478 0474 2
479 0475 2 | REMOVE command
480 0476 2
481 0477 2
482 0478 2
483 P 0479 2 $STATE (DO_REMOVE,
484 P 0480 2 ((UIC), MORE, ACT_REMOVE)
485 0481 2 );
486 0482 2
487 0483 2
488 0484 2 | Process additional commands on line
489 0485 2
490 0486 2
491 P 0487 2 $STATE (MORE,
492 P 0488 2 (':' START),
493 P 0489 2 (TPAS_EOS, TPAS_EXIT)
494 0490 2 );
495 0491 2
496 0492 2 | Process command switches
497 0493 2
498 0494 2
499 0495 2
500 P 0496 2 $STATE (CMD_SWIT,
501 P 0497 2 ('/' )
502 0498 2 );
503 0499 2
504 P 0500 2 $STATE (
505 P 0501 2 ('PERMQUOTA', DO_PERMQUOTA,, 1^PERM_SPEC, UIC_FLAGS),
506 P 0502 2 ('OVERDRAFT', DO_OVERDRAFT,, 1^OVER_SPEC, UIC_FLAGS),
507 P 0503 2 (TPAS_SYMBOL,, INV_SWITCH)
508 0504 2 );
509 0505 2
510 P 0506 2 $STATE (DO_PERMQUOTA,
511 P 0507 2 ('=')
512 0508 2 );
513 0509 2
514 P 0510 2 $STATE (
515 P 0511 2 (TPAS_DECIMAL, TPAS_EXIT..., PERM_VALUE)
516 0512 2 );
517 0513 2
518 P 0514 2 $STATE (DO_OVERDRAFT,
519 P 0515 2 ('=')
520 0516 2 );
521 0517 2
522 P 0518 2 $STATE (,

```

```

523 P 0519 2 (TPAS_DECIMAL, TPAS_EXIT..., OVER_VALUE)
524 0520 2 ;
525 0521 2 ;
526 0522 2 ;
527 0523 2 | Process device name
528 0524 2 ;
529 0525 2 ;
530 P 0526 2 SSTATE (DEV_SPEC,
531 0527 2 (TPAS_SYMBOL)
532 0528 2 );
533 0529 2 ;
534 P 0530 2 SSTATE {
535 0531 2 (':', TPAS_EXIT)
536 0532 2 (TPAS_LAMBDA, TPAS_EXIT)
537 0533 2 );
538 0534 2 ;
539 0535 2 ;
540 P 0536 2 | Process UIC
541 0537 2 ;
542 0538 2 ;
543 P 0539 2 SSTATE (UIC,
544 0540 2 (TPAS_IDENT, TPAS_EXIT..., UIC_VALUE)
545 0541 2 );
546 0542 2 ;
547 0543 2 ;
548 0544 2 | HELP command
549 0545 2 ;
550 0546 2 ;
551 P 0547 2 SSTATE (DO_HELP,
552 0548 2 (TPAS_STRING, DO_HELP, SAVE_KEY),
553 0549 2 ((DO_QUALIFIER), DO_HELP, SAVE_KEY),
554 0550 2 ('*', DO_HELP, SAVE_KEY),
555 0551 2 ((ELIPSIS), DO_HELP, SAVE_KEY),
556 0552 2 (TPAS_LAMBDA, MORE, ACT_HELP)
557 0553 2 );
558 0554 2 ;
559 P 0555 2 SSTATE (DO_QUALIFIER,
560 0556 2 ('/*')
561 0557 2 );
562 0558 2 ;
563 P 0559 2 SSTATE {
564 0560 2 ('PERMQUOTA', TPAS_EXIT),
565 0561 2 ('OVERDRAFT', TPAS_EXIT),
566 0562 2 (TPAS_STRING, TPAS_EXIT)
567 0563 2 );
568 0564 2 ;
569 P 0565 2 SSTATE (ELIPSIS,
570 0566 2 ('.')
571 0567 2 );
572 P 0568 2 SSTATE {
573 0569 2 ('..')
574 0570 2 );
575 0571 2 ;
576 0572 2 ;
577 P 0573 2 SSTATE {
578 0574 2 ('..', TPAS_EXIT)
579 0575 2 );

```

581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637

0576 2
0577 2
0578 2 | Set up a channel to the default disk, if it is defined.
0579 2
0580 2
0581 2 | ENABLE MAIN_HANDLER;
0582 2
0583 2
0584 2
0585 2
0586 2 | USE_DEFAULT ();
0587 2
0588 2
0589 2 | EXIT_HNDLR_DESC[XHNDLR_ADDRESS] = EXIT_HANDLER;
0590 2
0591 2
0592 2
0593 2
0594 2
0595 2 | EXIT_HNDLR_DESC[XHNDLR_ARGCNT] = 1;
0596 2
0597 2
0598 2
0599 2
0600 2
0601 2
0602 2
0603 2
0604 2
0605 2
0606 2
0607 2
0608 2
0609 2
0610 2
0611 2
0612 2
0613 2
0614 2
0615 2
0616 2
0617 2
0618 2
0619 2
0620 2
0621 2
0622 2
0623 2
0624 2
0625 2
0626 2
0627 2
0628 2
0629 2
P 0630 2
P 0631 2
P 0632 2

| EXIT_HNDLR_DESC[XHNDLR_STSADDR] = EXIT_HNDLR_DESC[XHNDLR_STSADDR+1];
SDCLEXH (DESBLK=EXIT_HNDLR_DESC);
| Acquire a command line, convert to upper case, and parse it. Command
processing is actually done by parser action routines. If a syntax error
occurs, output an error message. Errors occurring during the command
processsing are signalled at that time.
WHILE 1 DO
BEGIN
COMMAND_DESC[0] = COMMAND_LENGTH;
STATUS = LIBSGET_INPUT (COMMAND_DESC, DESCRIPTOR ('DISKQ>'));
IF NOT .STATUS
THEN
BEGIN
IF .STATUS NEQ RMSS_EOF
THEN ERR_MESSAGE (DSKQS_CMD_ERR, .STATUS);
RETURN 1;
END;
CHSTRANSFORM (UPCASE_TABLE, .COMMAND_DESC[0], .COMMAND_DESC[1], .0,
.COMMAND_DESC[0], .COMMAND_DESC[1]);
P = .COMMAND_DESC[0] + .COMMAND_DESC[1];
UNTIL CHSRCHAR (.P-1) NEQ ''
DO P = .P - 1;
.COMMAND_DESC[0] = .P - .COMMAND_DESC[1];
CHSFILL (0, ZERO_LENGTH, ZERO_AREA);
TPARSE_BLOCK[TPASL_STRINGCNT] = .COMMAND_DESC[0];
TPARSE_BLOCK[TPASL_STRINGPTR] = .COMMAND_DESC[1];
STATUS = LIBSTPARSE (TPARSE_BLOCK, STATE_TABLE, KEY_TABLE);
IF NOT .STATUS
THEN
BEGIN
IF .STATUS EQS LIBS_SYNTAXERR
THEN STATUS = DSKQS_SYNTAX;
ERR_MESSAGE (.STATUS,
.TPARSE_BLOCK[TPASL_TOKENPTR] - .COMMAND_DESC[1],
.COMMAND_DESC[1],

```

638 P 0633 6 .TPARSE_BLOCK[TPASL_TOKENCNT].
639 P 0634 4 .TPARSE_BLOCK[TPASL_TOKENPTR].
640 P 0635 4 .TPARSE_BLOCK[TPASL_STRINGCNT] - .TPARSE_BLOCK[TPASL_TOKENCNT].
641 P 0636 4 .TPARSE_BLOCK[TPASI_STRINGPTR] + .TPARSE_BLOCK[TPASL_TOKENCNT].
642 P 0637 4 )
643 P 0638 4 END:
644 P 0639 4
645 P 0640 4 IF .CLEANUP_FLAGS[CLF_EXIT]
646 P 0641 4 THEN RETURN 1 ! if EXIT command encountered
647 P 0642 4 ! then exit DISK_QUOTA
648 P 0643 4 END: ! end of command loop
649 P 0644 4
650 P 0645 2 1
651 P 0646 1 END: ! end of routine DISK_QUOTA

```

```

.TITLE DISKQUOTA
.IDENT \V04-000\
.PSECT _LIBSKEY1$,NOWRT, SHR, PIC,1

        00000 ;TPASKEYSTO
        U.68: .BLKB 0
        44 44 41 00000 ;TPASKEYST
        U.70: .ASCII \ADD\
        FF 00003 .BYTE -1
        00004 ;TPASKEYSTO
        U.74: .BLKB 0
        45 54 41 45 52 43 00004 ;TPASKEYST
        U.76: .ASCII \CREATE\
        FF 0000A .BYTE -1
        0000B ;TPASKEYSTO
        U.81: .BLKB 0
        45 4C 42 41 53 49 44 00008 ;TPASKEYST
        U.83: .ASCII \DISABLE\
        FF 00012 .BYTE -1
        00013 ;TPASKEYSTO
        U.87: .BLKB 0
        45 4C 42 41 4E 45 00013 ;TPASKEYST
        U.89: .ASCII \ENABLE\
        FF 00019 .BYTE -1
        0001A ;TPASKEYSTO
        U.93: .BLKB 0
        54 49 58 45 0001A ;TPASKEYST
        U.95: .ASCII \EXIT\
        FF 0001E .BYTE -1
        0001F ;TPASKEYSTO
        U.100: .BLKB 0
        50 4C 45 48 0001F ;TPASKEYST
        U.102: .ASCII \HELP\
        FF 00023 .BYTE -1
        00024 ;TPASKEYSTO
        U.106: .BLKB 0
        59 46 49 44 4F 4D 00024 ;TPASKEYST
        U.108: .ASCII \MODIFY\
        FF 0002A .BYTE -1
        0002B ;TPASKEYSTO

```

44 4C 49 55 42 45 52	0002B	U.112: ;TPASKEY\$T	BLKB	0	
	FF 00032	U.114: .ASCII	\REBUILD\		
	00033	BYTE	-1		
45 56 4F 4D 45 52	00033	;TPASKEY\$TO			
	U.120: ;TPASKEY\$T	BLKB	0		
	FF 00039	U.124: .ASCII	\REMOVE\		
	0003A	BYTE	-1		
57 4F 48 53	0003A	;TPASKEY\$T	BLKB	0	
	U.126: .ASCII	\SHOW\			
	FF 0003E	BYTE	-1		
	0003F	;TPASKEY\$TO			
45 53 55	0003F	U.130: ;TPASKEY\$T	BLKB	0	
	FF 00042	U.132: .ASCII	\USE\		
	FF 00043	BYTE	-1		
	00044	;TPASKEY\$ILL			
	U.140: .BYTE	-1			
	00044	;TPASKEY\$TO			
41 54 4F 55 51 4D 52 45 50	00044	U.187: ;TPASKEY\$T	BLKB	0	
	FF 0004D	U.189: .ASCII	\PERMQUOTA\		
	0004E	BYTE	-1		
54 46 41 52 44 52 45 56 4F	0004E	;TPASKEY\$T	BLKB	0	
	FF 00057	U.197: .ASCII	\OVERDRAFT\		
	FF 00058	BYTE	-1		
	00059	;TPASKEY\$ILL			
	U.205: .BYTE	-1			
	00059	;TPASKEY\$TO			
41 54 4F 55 51 4D 52 45 50	00059	U.242: ;TPASKEY\$T	BLKB	0	
	FF 00062	U.244: .ASCII	\PERMQUOTA\		
	00063	BYTE	-1		
54 46 41 52 44 52 45 56 4F	00063	;TPASKEY\$T	BLKB	0	
	FF 0006C	U.249: .ASCII	\OVERDRAFT\		
	FF 0006D	;TPASKEY\$ILL			
	U.254: .BYTE	-1			
		.PSECT	_LIBSSTATES,NOWRT, SHR, PIC,1		
		00000 STATE_TABLE::			
	1100 00000	START: ;TPASTYPE	BLKB	0	
	00000	WORD			
	0000* 00002	;TPASTARGET	4352		
	9101 00004	WORD	<<U.72-U.73>-2>		
		;TPASTYPE			
		WORD			
		U.77: -28415			

00000000V 00006	;TPSACTION	
0000* 0000A	U.78: .LONG	<<ACT_CREATE-U.78>-4>
9102 0000C	U.80: .WORD	<<U.79-U.80>-2>
00000000V 0000E	U.84: .WORD	-28414
0000* 00012	U.85: .LONG	<<ACT_DISABLE-U.85>-4>
9103 00014	U.86: .WORD	<<U.79-U.86>-2>
00000000V 00016	U.90: .WORD	-28413
0000* 0001A	U.91: .LONG	<<ACT_ENABLE-U.91>-4>
7104 0001C	U.92: .WORD	<<U.79-U.92>-2>
00000000* 0001E	U.96: .WORD	28932
00000002 00022	U.97: .LONG	<<CLEANUP_FLAGS-U.97>-4>
FFFF 00026	U.98: .LONG	2
1105 00028	U.99: .WORD	-1
0000* 0002A	U.103: .WORD	4357
1106 0002C	U.105: .WORD	<<U.104-U.105>-2>
0000* 0002E	U.109: .WORD	4358
9107 00030	U.111: .WORD	<<U.110-U.111>-2>
00000000V 00032	U.115: .WORD	-28409
0000* 00036	U.116: .LONG	<<ACT_REBUILD-U.116>-4>
1108 00038	U.117: .WORD	<<U.79-U.117>-2>
0000* 0003A	U.121: .WORD	4360
1109 0003C	U.123: .WORD	<<U.122-U.123>-2>
0000* 0003E	U.127: .WORD	4361
110A 00040	U.129: .WORD	<<U.128-U.129>-2>
0000* 00042	U.133: .WORD	4362
81F1 00044	U.135: .WORD	<<U.134-U.135>-2>
00000000V 00046	U.136: .WORD	-32271
15F7 0004A	U.137: .LONG	<<INV_COMMAND-U.137>-4>
FFFF 0004C	U.138: .WORD	5623

L 10
15-Sep-1984 23:38:38
14-Sep-1984 12:19:46VAX-11 Bliss-32 V4.0-742
[DISKQ.SRC]DISKQUOTA.B32;1Page 17
(6)

J004E U.139: .WORD -1 ;
9DF8 0004E ;DO USE ;
U.134: .BLKB 0 ;
0000* 00050 ;TPASSTYPÉ ;
U.141: .WORD -25096 ;
00000000V 00052 ;TPASACTION ;
U.143: .WORD <<U.142-U.143>-2> ;
0000* 00056 ;TPASTARGET ;
U.144: .LONG <<ACT_USE-U.144>-4> ;
00058 U.145: .WORD <<U.79-U.145>-2> ;
;DO ADD ;
19F8 00058 U.72: .BLKB 0 ;
0000* 0005A ;TPASSUBEXP ;
U.146: .WORD 6648 ;
0000* 0005C ;TPASTARGET ;
U.148: .WORD <<U.147-U.148>-2> ;
1DF8 0005E ;TPASSTYPÉ ;
U.149: .WORD <<U.72-U.149>-2> ;
0G00* 00060 ;TPASSUBEXP ;
U.150: .WORD 7672 ;
0000* 00062 ;TPASTARGET ;
U.152: .WORD <<U.151-U.152>-2> ;
00064 U.154: .WORD <<U.153-U.154>-2> ;
;DO ADD1 ;
19F8 00064 U.153: .BLKB 0 ;
0000* 00066 ;TPASSTYPÉ ;
U.155: .WORD 6648 ;
0000* 00068 ;TPASTARGET ;
U.156: .WORD <<U.147-U.156>-2> ;
95F6 0006A ;TPASSTYPÉ ;
U.157: .WORD <<U.153-U.157>-2> ;
00000000V 0006C ;TPASACTION ;
U.158: .WORD -27146 ;
0000* 00070 ;TPASTARGET ;
U.159: .LONG <<ACT_ADD-U.159>-4> ;
00072 ;DO MODIFY ;
U.160: .WORD <<U.79-U.160>-2> ;
19F8 00072 U.1T0: .BLKB 0 ;
0000* 00074 ;TPASSUBEXP ;
U.161: .WORD 6648 ;
0000* 00076 ;TPASTARGET ;
U.162: .WORD <<U.147-U.162>-2> ;
1DF8 00078 ;TPASSTYPÉ ;
U.163: .WORD <<U.110-U.163>-2> ;
0000* 0007A ;TPASSUBEXP ;
U.164: .WORD 7672 ;
0000* 0007C ;TPASTARGET ;
U.165: .WORD <<U.151-U.165>-2> ;
0007E ;DO MODIFY1 ;
U.166: .BLKB 0 ;
19F8 0007E ;TPASSTYPÉ ;
U.168: .WORD 6648 ;

D1
VC

0000* 00080 ;TPASSUBEXP
 U.169: .WORD <<U.147-U.169>-2>
0000* 00082 ;TPASTARGET
 U.170: .WORD <<U.166-U.170>-2>
95F6 00084 ;TPASTYPE
 U.171: .WORD -27146
00000000V 00086 ;TPASACTION
 U.172: .LONG <<ACT_MODIFY-U.172>-4>
0000* 0008A ;TPASTARGET
 U.173: .WORD <<U.79-U.173>-2>
 0008C ;DO SHOW
 U.178: .BLKB 0
9DF8 0008C ;TPASTYPE
 U.174: .WORD -25096
0000* 0008E ;TPASSUBEXP
 U.175: .WORD <<U.151-U.175>-2>
00000000V 00090 ;TPASACTION
 U.176: .LONG <<ACT_SHOW-U.176>-4>
0000* 00094 ;TPASTARGET
 U.177: .WORD <<U.79-U.177>-2>
 00096 ;DO REMOVE
 U.172: .BLKB 0
9DF8 00096 ;TPASTYPE
 U.178: .WORD -25096
0000* 00098 ;TPASSUBEXP
 U.179: .WORD <<U.151-U.179>-2>
00000000V 0009A ;TPASACTION
 U.180: .LONG <<ACT_REMOVE-U.180>-4>
0000* 0009E ;TPASTARGET
 U.181: .WORD <<U.79-U.181>-2>
 000A0 ;MORE
 U.79: .BLKB 0
103B 000A0 ;TPASTYPE
 U.182: .WORD 4155
0000* 000A2 ;TPASTARGET
 U.183: .WORD <<START-U.183>-2>
15F7 000A4 ;TPASTYPE
 U.184: .WORD 5623
FFFF 000A6 ;TPASTARGET
 U.185: .WORD -1
 000A8 ;CMD_SWIT
 U.147: .BLKB 0
042F 000A8 ;TPASTYPE
 U.186: .WORD 1071
7108 000AA ;TPASTYPE
 U.190: .WORD 28939
00000000* 000AC ;TPASADDR
 U.191: .LONG <<UI_C_FLAGS-U.149>-4>
00000008 000B0 ;TPASMASK
 U.192: .LONG 8
0000* 000B4 ;TPASTARGET
 U.194: .WORD <<U.193-U.194>-2>
710C 000B6 ;TPASTYPE
 U.198: .WORD 28940
00000000* 000B8 ;TPASADDR
 U.199: .LONG <<UI_C_FLAGS-U.199>-4>
00000010 000BC ;TPASMASK

0000* 000C0 U.200: .LONG 16
85F1 000C2 ;TPA\$TARGET
U.202: .WORD <<U.201-U.202>-2>
00000000V 000C4 ;TPA\$ACTION
U.203: .WORD -31247
000CB ;DO PERMQUOTA
U.193: .BLKB 0
043D 000CB ;TPA\$TYPE
U.206: .WORD 1085
55F3 000CA ;TPA\$TYPE
U.207: .WORD 22003
00000000* 000CC ;TPA\$ADDR
U.208: .LONG <<PERM_VALUE-U.208>-4>
FFFF 000D0 ;TPA\$TARGET
U.209: .WORD -1
000D2 ;DO OVERDRAFT
U.201: .BLKB 0
043D 000D2 ;TPA\$TYPE
U.210: .WORD 1085
55F3 000D4 ;TPA\$TYPE
U.211: .WORD 22003
00000000* 000D6 ;TPA\$ADDR
U.212: .LONG <<OVER_VALUE-U.212>-4>
FFFF 000DA ;TPA\$TARGET
U.213: .WORD -1
000DC ;DEV SPEC
U.142: .BLKB 0
05F1 000DC ;TPA\$TYPE
U.214: .WORD 1521
103A 000Df ;TPA\$TYPE
U.215: .WORD 4154
FFFF 000E0 ;TPA\$TARGET
U.216: .WORD -1
15F6 000E2 ;TPA\$TYPE
U.217: .WORD 5622
FFFF 000E4 ;TPA\$TARGET
U.218: .WORD -1
000E6 ;UIC
U.151: .BLKB 0
55EC 000E6 ;TPA\$TYPE
U.219: .WORD 21996
00000000* 000E8 ;TPA\$ADDR
U.220: .LONG <<UIC_VALUE-U.220>-4>
FFFF 000Ec ;TPA\$TARGET
U.221: .WORD -1
000EE ;DO HELP
U.104: .BLKB 0
91F0 000EE ;TPA\$TYPE
U.222: .WORD -28176
00000000V 000F0 ;TPA\$ACTION
0000* 000F4 ;TPA\$TARGET
99F8 000F6 ;TPA\$TYPE
U.223: .LONG <<SAVE_KEY-U.223>-4>
U.224: .WORD <<U.104-U.224>-2>
U.225: .WORD -26120

0000* 000F8 :TPASSUBEXP
00000000V 000FA ;TPASACTION
0000* 000FE ;TPASTARGET
902A 00100 ;TPASTYPE
00000000V 00102 ;TPASACTION
0000* 00106 ;TPASTARGET
99F8 00108 ;TPASTYPE
0000* 0010A ;TPASSUBEXP
00000000V 0010C ;TPASACTION
0000* 00110 ;TPASTARGET
95F6 00112 ;TPASTYPE
00000000V 00114 ;TPASACTION
0000* 00118 ;TPASTARGET
0011A ;DO QUALIFIER
042F 0011A ;TPASTYPE
110D 0011C ;TPASTYPE
FFFF 0011E ;TPASTARGET
110E 00120 ;TPASTYPE
FFFF 00122 ;TPASTARGET
15F0 00124 ;TPASTYPE
FFFF 00126 ;TPASTARGET
00128 ;ELIPSIS
042E 00128 ;TPASTYPE
042E 0012A ;TPASTYPE
142E 0012C ;TPASTYPE
FFFF 0012E ;TPASTARGET
U.227: WORD <<U.226-U.227>-2>
U.228: LONG <<SAVE_KEY-U.228>-4>
U.229: WORD <<U.104-U.229>-2>
U.230: WORD -28630
U.231: LONG <<SAVE_KEY-U.231>-4>
U.232: WORD <<U.104-U.232>-2>
U.233: WORD -26120
U.235: WORD <<U.234-U.235>-2>
U.236: LONG <<SAVE_KEY-U.236>-4>
U.237: WORD <<U.104-U.237>-2>
U.238: WORD -27146
U.239: LONG <<ACT_HELP-U.239>-4>
U.240: WORD <<U.79-U.240>-2>
U.226: BLKB 0
U.241: WORD 1071
U.245: WORD 4365
U.246: WORD -1
U.250: WORD 4366
U.251: WORD -1
U.252: WORD 5616
U.253: WORD -1
U.234: BLKB 0
U.255: WORD 1070
U.256: WORD 1070
U.257: WORD 5166
U.258: WORD -1
.PSECT _LIB\$KEYOS,NOWRT, SHR, PIC,1
00000 KEY_TABLE::
.BLKB 0

00000 ;TPASKEY0
U.67: .BLKB 0
0000* 00000 ;TPASKEY
U.69: .WORD <U.68-U.67>
0000* 00002 ;TPASKEY
U.75: .WORD <U.74-U.67>
0000* 00004 ;TPASKEY
U.82: .WORD <U.81-U.67>
0000* 00006 ;TPASKEY
U.88: .WORD <U.87-U.67>
0000* 00008 ;TPASKEY
U.94: .WORD <U.93-U.67>
0000* 0000A ;TPASKEY
U.101: .WORD <U.100-U.67>
0000* 0000C ;TPASKEY
U.107: .WORD <U.106-U.67>
0000* 0000E ;TPASKEY
U.113: .WORD <U.112-U.67>
0000* 00010 ;TPASKEY
U.119: .WORD <U.118-U.67>
0000* 00012 ;TPASKEY
U.125: .WORD <U.124-U.67>
0000* 00014 ;TPASKEY
U.131: .WORD <U.130-U.67>
0000* 00016 ;TPASKEY
U.188: .WORD <U.187-U.67>
0000* 00018 ;TPASKEY
U.196: .WORD <U.195-U.67>
0000* 0001A ;TPASKEY
U.243: .WORD <U.242-U.67>
0000* 0001C ;TPASKEY
U.248: .WORD <U.247-U.67>
.PSECT SMSG_INDEX,NOWRT,NOEXE,2

00000 MESSAGE_TABLE:
00000000' 00000 ;MSG_INDEX
U.2: .BLKB 0
00000000' 00004 ;MSG_INDEX
U.4: .ADDRESS U.1
00000000' 00008 ;MSG_INDEX
U.6: .ADDRESS U.3
00000000' 0000C ;MSG_INDEX
U.8: .ADDRESS U.5
00000000' 00010 ;MSG_INDEX
U.10: .ADDRESS U.7
00000000' 00014 ;MSG_INDEX
U.12: .ADDRESS U.9
00000000' 00018 ;MSG_INDEX
U.14: .ADDRESS U.11
00000000' 0001C ;MSG_INDEX
U.16: .ADDRESS U.13
00000000' 00020 ;MSG_INDEX
U.18: .ADDRESS U.15
00000000' 00024 ;MSG_INDEX
U.20: .ADDRESS U.17

00000000' 00028 ;MSG_INDEX
00000000' 0002C ;MSG_INDEX U.22: ADDRESS U.21
00000000' 00030 ;MSG_INDEX U.24: ADDRESS U.23
00000000' 00034 ;MSG_INDEX U.26: ADDRESS U.25
00000000' 00038 ;MSG_INDEX U.28: ADDRESS U.27
00000000' 0003C ;MSG_INDEX U.30: ADDRESS U.29
00000000' 00040 ;MSG_INDEX U.32: ADDRESS U.31
00000000' 00044 ;MSG_INDEX U.34: ADDRESS U.33
00000000' 00048 ;MSG_INDEX U.36: ADDRESS U.35
00000000' 0004C ;MSG_INDEX U.38: ADDRESS U.37
00000000' 00050 ;MSG_INDEX U.40: ADDRESS U.39
00000000' 00054 ;MSG_INDEX U.42: ADDRESS U.41
00000000' 00058 ;MSG_INDEX U.44: ADDRESS U.43
00000000' 0005C ;MSG_INDEX U.46: ADDRESS U.45
00000000' 00060 ;MSG_INDEX U.48: ADDRESS U.47
00000000' 00064 ;MSG_INDEX U.50: ADDRESS U.49
00000000' 00068 ;MSG_INDEX U.52: ADDRESS U.51
00000000' 0006C ;MSG_INDEX U.54: ADDRESS U.53
00000000' 00070 ;MSG_INDEX U.56: ADDRESS U.55
00000000' 00074 ;MSG_INDEX U.58: ADDRESS U.57
00000000' 00078 ;MSG_INDEX U.60: ADDRESS U.59
00000000' 0007C ;MSG_INDEX U.62: ADDRESS U.61
00000000' 00080 ;MSG_INDEX U.64: ADDRESS U.63
00000000' 00084 ;MSG_INDEX U.66: ADDRESS U.65

.PSECT \$MSG_TEXT,NOWRT,NOEXE,0

00000 MESSAGE_TEXT:
2C 00 00000 ;MSG_TEXT BLKB 0
2D 51 48 53 49 44 25 00002 U.1: .BYTE 0, 44
46 00009 .ASCII \\$DISKQ-\
2D 0000A .ASCII \F\
52 52 45 5F 44 40 43 0000B .ASCII \-\
52 .ASCII \CMD_ERR\

69	64	61	65	72	20	72	6F	72	6D	65	20	4F	20	2F	2C	00012	.ASCII	\\$I/O error reading commands\
				73	64	6E	61	6D	6F	63	20		67	6E	00014	.ASCII		
													33	06	00023	.ASCII		
															0002E	.ASCII		
															00030	:MSG_TEXT		
															U.3:	BLKB	2	
																.BYTE	6, 51	
																.ASCII	\\$DISKQ-\	
																.ASCII	\E\	
																.ASCII	\-\	
																.ASCII	\INV_CMD\	
																.ASCII	\.\	
																.ASCII	\unrecognized command!\!AD\<92>\!AD\<92>	
6F	63	20	64	65	7A	69	6E	67	6F	63	65	72	6E	75	00044	.ASCII		
5C	44	41	21	5C	44	41	21	2F	21	64	6E	61	6D	60	00053	.ASCII		
												44	41	21	00062	.ASCII	\!AD\	
															00065	.BLKB	3	
																.BYTE	6, 48	
																.ASCII	\\$DISKQ-\	
																.ASCII	\E\	
																.ASCII	\-\	
																.ASCII	\AMB_CMD\	
																.ASCII	\.\	
61	6D	6D	63	20	73	75	6F	75	67	69	62	6D	61	0007C	.ASCII	\ambiguous command!\!AD\<92>\!AD\<92>\!\		
21	5C	44	41	21	5C	44	41	21	2F	21	64	6E	0008B	.ASCII				
												44	41	00098	.ASCII	\AD\		
															0009A	.BLKB	2	
																.BYTE	6, 54	
																.ASCII	\\$DISKQ-\	
																.ASCII	\E\	
																.ASCII	\-\	
																.ASCII	\INV_QUAL\	
																.ASCII	\.\	
75	71	20	64	65	7A	69	6E	67	6F	63	65	72	6E	75	000B1	.ASCII	\unrecognized qualifier!\!AD\<92>\!AD\<92>	
41	21	5C	44	41	21	2F	21	72	65	69	66	69	6C	61	000C0	.ASCII		
																.ASCII	000CF	
																.ASCII	000D1	
															33	06	000D4	
																:MSG_TEXT		
																U.9:		
																.ASCII	\!AD\	
																.BYTE	6, 51	
																.ASCII	\\$DISKQ-\	
																.ASCII	\E\	
																.ASCII	\-\	
																.ASCII	\AMB_QUAL\	
																.ASCII	\.\	
69	6C	61	75	71	20	73	75	6F	75	67	69	62	6D	61	000E9	.ASCII	\ambiguous qualifier!\!AD\<92>\!AD\<92>	
5C	44	41	21	5C	44	41	21	2F	21	72	65	69	66	000F8	.ASCII			
																.ASCII	00106	
																.BLKB	3	
																.BYTE	6, 51	
																.ASCII	\\$DISKQ-\	
																.ASCII	\E\	
																.ASCII	\-\	
																.ASCII	\INV_UIC\	
																.ASCII	\.\	
41	21	2F	21	43	49	55	20	64	69	6C	61	76	6E	69	00120	.ASCII	\invalid UIC!\!AD\<92>\!AD\<92>\!AD\	
																.BLKB	3	
																.BYTE	6, 42	
																.ASCII	\\$DISKQ-\	
																.ASCII	\E\	
																.ASCII	\-\	
																.ASCII	\INV_UIC\	
																.ASCII	\.\	
																.ASCII	\invalid UIC!\!AD\<92>\!AD\<92>\!AD\	

7A	69	6C	61	69	74	69	61	74	6F	70	75	71	20	72	72	65	0024E	.ASCII	\.\`		
	65	6C	69	66	20	61	74	6F	70	75	71	20	72	72	65	00250	.ASCII	\error initializing quota file\			
																0025F	BLKB	3			
																00260					
																00270	:MSG_TEXT				
																U.25:					
																	.BYTE	0 42			
																	.ASCII	\\$DISKQ-\`			
																	.ASCII	\E\`			
																	.ASCII	\-\`			
																	.ASCII	\CLOSEERR\`			
																	.ASCII	\.\`			
71	20	67	6E	69	73	6F	6C	63	20	72	6F	71	72	72	65	00284	.ASCII	\error closing quota file\`			
	65	6C	69	66	20	61	74	72	6F	75	71	6F	75	00293	2C	0029C	:MSG_TEXT				
																	U.27:				
																	.BYTE	0 44			
																	.ASCII	\\$DISKQ-\`			
																	.ASCII	\E\`			
																	.ASCII	\-\`			
																	.ASCII	\ACTERR\`			
6C	62	61	6E	65	20	6F	74	20	61	74	6F	75	71	20	61	66	002AF	.ASCII	\failed to enable quota file\`		
	65	6C	69	66	20	61	74	6F	75	71	20	65	002BE	2C	002CA	:MSG_TEXT					
																	BLKB	2			
																	.BYTE	0 46			
																	.ASCII	\\$DISKQ-\`			
																	.ASCII	\E\`			
																	.ASCII	\-\`			
62	61	73	69	64	20	6F	74	20	64	65	6C	71	69	61	66	002E0	.ASCII	\failed to disable quota file\`			
	65	6C	69	66	20	61	74	6F	75	71	20	65	002EF	2F	002FC	:MSG_TEXT					
																	U.31:				
																	.BYTE	0 47			
																	.ASCII	\\$DISKQ-\`			
																	.ASCII	\E\`			
																	.ASCII	\-\`			
																	.ASCII	\DACTERR\`			
71	20	64	66	61	20	6F	74	20	64	65	6C	69	61	66	0030F	.ASCII	\failed to add quota file entry\`				
	79	72	74	6E	65	20	65	6C	69	66	20	61	74	6F	75	0031E	34	00	00330	:MSG_TEXT	
																	BLKB	3			
																	.BYTE	0 52			
																	.ASCII	\\$DISKQ-\`			
																	.ASCII	\E\`			
																	.ASCII	\-\`			
																	.ASCII	\ADDERR\`			
																	.ASCII	\-\`			
71	20	64	66	61	20	6F	74	20	64	65	6C	69	61	66	0030F	.ASCII	\failed to add quota file entry\`				
	79	72	74	6E	65	20	65	6C	69	66	20	61	74	6F	75	0031E	34	00	00330	:MSG_TEXT	
																	BLKB	3			
																	.BYTE	0 52			
																	.ASCII	\\$DISKQ-\`			
																	.ASCII	\E\`			
																	.ASCII	\-\`			
																	.ASCII	\REMOVERR\`			
76	6F	6D	65	72	20	6F	74	20	64	65	6C	69	61	66	00345	.ASCII	\failed to remove quota file entry\`				
	6E	65	20	65	6C	69	66	20	61	74	6F	75	71	20	65	00354	35	00	00368	:MSG_TEXT	
																	BLKB	2			
																	.BYTE	0 53			
																	.ASCII	\\$DISKQ-\`			
																	.ASCII	\E\`			
																	.ASCII	\-\`			
																	.ASCII	\-\`			

65	60	75	6C	6F	76	6C	64	79	6E	61	68	6D	20	6F	74	6F	20	74	0048B	.ASCII	\e\										
																		65	0049A	.BLKB	3										
																	46	01	004A8	:MSG_TEXT											
																	46	01	U.47:	.BYTE	1 70										
																	2D	51	48	53	49	44	25	004AA	.ASCII	\\$DISK0-1					
																	45	004B1	.ASCII	\E\											
																	2D	41	004B2	.ASCII	\-\										
																	20	2C	004B3	.ASCII	\ACCINDEXF\										
																	20	2C	004BC	.ASCII	\-\										
																	73	004CD	.ASCII	\failed to access index file on relative \											
73	65	63	63	61	20	6F	74	20	64	65	6C	69	61	66	004BE	.ASCII	\volume !UW\														
6E	6F	20	65	6C	69	66	20	78	65	64	6E	69	20	73	004CD	.ASCII															
																	57	55	21	20	65	60	75	6C	69	20	72	20	004DC	.ASCII	\volume !UW\
																	6F	004E6	.ASCII												
																	2E	00	004F0	:MSG_TEXT											
																	2D	51	48	53	49	44	25	004F2	.BYTE	0 66					
																	45	004F9	.ASCII	\\$DISK0-1											
																	2D	41	004FA	.ASCII	\E\										
																	20	2C	004FB	.ASCII	\-\										
																	73	00503	.ASCII	\ACCQFILE\											
																	61	66	00505	.ASCII	\-\										
																	20	73	00514	.ASCII	\failed to access quota file\										
73	65	63	63	61	20	6F	74	20	64	65	6C	69	61	66	00520	:MSG_TEXT															
65	6C	69	66	20	61	74	6F	75	71	20	73	00514	.ASCII																		
																	30	00	00520	U.51:	.BYTE	0 48									
																	2D	51	48	53	49	44	25	00522	.ASCII	\\$DISK0-1					
																	45	00529	.ASCII	\E\											
																	2D	0052A	.ASCII	\-\											
																	52	52	45	52	41	54	4F	55	51	0052B	.ASCII	\QUOTARERR\			
																	20	2C	00534	.ASCII	\-\										
																	67	6E	00536	.ASCII	\\$/0 error reading quota file\										
69	64	61	65	72	20	72	6F	72	72	65	20	4F	2F	49	00545	.ASCII															
65	6C	69	66	20	61	74	6F	75	71	20	67	6E	00552	:MSG_TEXT																	
																	4E	01	00554	U.52:	.BYTE	2									
																	2D	51	48	53	49	44	25	00556	.ASCII	\\$DISK0-1					
																	45	0055D	.ASCII	\E\											
																	2D	0055E	.ASCII	\-\											
																	52	52	45	50	41	4D	54	49	42	0055F	.ASCII	\BITMAPERR\			
																	20	2C	00568	.ASCII	\-\										
																	67	6E	0056A	.ASCII	\\$/0 error reading index file bitmap on r\										
69	64	61	65	72	20	72	6F	72	72	65	20	4F	2F	49	00579	.ASCII															
62	20	65	6C	69	66	20	78	65	64	6E	69	20	67	6E	00579	:MSG_TEXT															
																	72	20	70	61	6D	74	69	00588	.ASCII	\relative volume !UW\					
20	65	60	75	6C	6F	76	20	65	76	69	74	61	65	00592	.ASCII																
																	57	55	21	4C	02	005A1	:MSG_TEXT								
																	4E	02	005A4	U.53:	.BYTE	2 76									
																	2D	51	48	53	49	44	25	005A6	.ASCII	\\$DISK0-1					
																	45	005AD	.ASCII	\W\											
																	2D	005AE	.ASCII	\-\											
																	52	52	45	52	45	44	41	65	48	005AF	.ASCII	\HEADERERR\			
																	20	2C	005B8	.ASCII	\-\										
																	67	6E	005BA	.ASCII	\\$/0 error reading file header !UL on rel\										
69	64	61	65	72	20	72	6F	72	72	65	20	4F	2F	49	005C9	:MSG_TEXT															
20	72	65	64	61	65	68	20	65	6C	69	66	20	67	6E	005D8	.ASCII															
																	76	20	6F	76	20	65	76	69	74	61	005E2	.ASCII	\relative volume !UW\		

1D	1C	1B	1A	19	18	17	16	15	14	13	12	11	10	0F	0000F
2C	2A	29	28	27	26	25	24	23	22	21	20	1F	1E	0001E	
3B	3A	39	38	37	36	35	34	33	32	31	30	2F	2E	0002D	
4A	49	48	47	46	45	44	43	42	41	40	3F	3E	3D	0003C	
59	58	57	56	55	54	53	52	51	50	4F	4E	4D	4C	0004B	
48	47	46	45	44	43	42	41	40	3F	3E	3D	3C	3B	0005A	
57	56	55	54	53	52	51	50	4F	4E	4D	4C	4B	4A	00069	
06	05	04	03	02	01	00	0F	0E	0D	0C	0B	0A	09	08	
15	14	13	12	11	10	0F	0E	0D	0C	0B	0A	09	08	07	
24	23	22	21	20	1F	1E	1D	1C	1B	1A	19	18	17	16	
33	32	31	30	2F	2E	2D	2C	2B	2A	29	28	27	26	25	
42	41	40	3F	3E	3D	3C	3B	3A	39	38	37	36	35	000B4	
51	50	4F	4E	4D	4C	4B	4A	49	48	47	46	45	44	000C3	
60	5F	5E	5D	5C	5B	5A	59	58	57	56	55	54	53	000D2	
4F	4E	4D	4C	4B	4A	49	48	47	46	45	44	43	42	000E1	
7E	7D	7C	7B	5A	59	58	57	56	55	54	53	52	51	50	
													7F	000FF	

13	14	15	16	17	18	19	20	21	22	-	-	-	-	-
23	24	25	26	27	28	29	30	31	32	-	-	-	-	-
33	34	35	36	37	38	39	40	41	42	-	-	-	-	-
43	44	45	46	47	48	49	50	51	52	-	-	-	-	-
53	54	55	56	57	58	59	60	61	62	-	-	-	-	-
63	64	65	66	67	68	69	70	71	72	-	-	-	-	-
73	74	75	76	77	78	79	80	81	82	-	-	-	-	-
83	84	85	86	87	88	89	90	91	92	-	-	-	-	-
93	94	95	96	97	98	99	100	101	102	-	-	-	-	-
0	1	2	3	4	5	6	7	8	9	-	-	-	-	-
123	124	125	126	127	128	129	130	131	132	-	-	-	-	-
5	6	7	8	9	10	11	12	13	14	15	-	-	-	-
16	17	18	19	20	21	22	23	24	25	-	-	-	-	-
26	27	28	29	30	31	32	33	34	35	-	-	-	-	-
36	37	38	39	40	41	42	43	44	45	-	-	-	-	-
46	47	48	49	50	51	52	53	54	55	-	-	-	-	-
56	57	58	59	60	61	62	63	64	65	-	-	-	-	-
66	67	68	69	70	71	72	73	74	75	-	-	-	-	-
76	77	78	79	80	81	82	83	84	85	-	-	-	-	-
86	87	88	89	90	91	92	93	94	95	-	-	-	-	-
96	65	66	67	68	69	70	71	72	73	-	-	-	-	-
74	75	76	77	78	79	80	81	82	83	-	-	-	-	-
84	85	86	87	88	89	90	123	124	-	-	-	-	-	-
125	126	127	-	-	-	-	-	-	-	-	-	-	-	-

3E 51 48 53 49 46 00100 P.AAE: .ASCII \DISKQ>\
 00106 .BLKB 2
 00000006 00108 P.AAD: .LONG 6
 00000000 0010C .ADDRESS P.AAE
 .PSECT \$0WN\$,\$0EXE,2
 00000 CHANNEL:.BLKB 2
 00002 .BLKB 2
 00004 IO_STATUS:
 .BLKB 8
 0000C COMMAND_LINE:
 .BLKB 132
 00090 OUTPUT_LINE:
 .BLKB 132
 00009084 00114 COMMAND_DESC:
 .LONG 132
 00000000 00118 .ADDRESS COMMAND_LINE
 00000084 0011C OUTPUT_DESC:
 .LONG 132
 00000000 00120 .ADDRESS OUTPUT_LINE
 .BCKB 20
 00124 EXIT_HNDLR_DESC:
 .BCKB 20
 00138 ZERO_AREA:
 .BLKB 0
 00138 CLEANUP_FLAGS:
 .BLKB 4
 0013C SRC_REC:.BLKB 32
 0015C DST_REC:.BLKB 32
 0017C QUOTA_FIB:
 .BLKB 64
 001BC UIC_FLAGS:

		001C0 KEY_VECTOR: .BLKB 4	
		001FB KEY_INDEX: .BLKB 56	
		001FC ZERO_END: .BLKB 4	
		00000020 001FC SRCREC_DESC: .BLKB 0	
		00000000' 00200 .LONG 32	
		00000020 00204 DSTREC_DESC: .ADDRESS SRC_REC	
		00000000' 00208 .LONG 32	
		00000040 0020C QFIB_DESC: .ADDRESS DST_REC	
		00000000' 00210 .LONG 64	
		00000002 00000008 00214 TPARSE_BLOCK: .ADDRESS QUOTA_FIB	
		00000000' 0021C .LONG 8 2	
31 38 53 59 53 2E 41 54 4F 55 51		00238 P.AAB: .ASCII \QUOTA.SYS;1\	
		00243 .BLKB 1	
		00000008 00244 P.AAA: .LONG 11	
		00000000' 00248 .ADDRESS P.AAB	

UIC_VALUE=	SRC_REC+4
PERR_VALUE=	SRC_REC+12
OVER_VALUE=	SRC_REC+16
QFILE_NAME=	P.AAA
UPCASE_TABLE=	P.AAC
.EXTRN LIB\$SYNTAXERR, LIB\$GET_INPUT	
.EXTRN LIB\$TPARSE, SYSSDCLEXH	
.PSECT SCODES,NOWRT,2	

	OFFC 00000	.ENTRY	DISK QUOTA, Save R2,R3,R4,R5,R6,R7,R8,R9,-	: 0335
		MOVAB	R10, R11	
		LIB\$SIGNAL, R11		
		COMMAND_DESC, R10		
		MOVAL	98. (FP)	0374
		CALLS	#0. USE DEFAULT	0583
		MOVAB	EXIT_HDLR, EXIT_HDLR_DESC+4	0589
		MOVL	#1 EXIT_HDLR_DESC+8	0590
		MOVAB	#1 EXIT_HDLR_DESC+16, EXIT_HDLR_DESC+12	0591
		PUSHAB	EXIT_HDLR_DESC	0593
		CALLS	#1 SYSSDCLEXH	0604
		MOVZBL	#132, COMMAND_DESC	0605
		PUSHAB	P.AAD	
		PUSHL	R10	
		CALLS	#2. LIB\$GET_INPUT	
		MOVL	R0, STATUS	
		BLBS	STATUS, 38	0606
		CMPL	STATUS, #98938	0609
		BEGL	28	
		PUSHL	STATUS	0610
		PUSHL	#4521984	
		CALLS	#2. LIB\$SIGNAL	
		BRW	88	0611

00000000 EF	00	56	6A	7D	00067	38:	MOVQ	COMMAND_DESC, R6	0614
		67	2E	0006A			MOVTC	R6, (R7), #0, UPCASE_TABLE, R6, (R7)	0615
	58	56	57	C1	00073		ADDL3	R7, R6, P	0616
		20	57	A8	00075	48:	CMPB	-1(P), #32	0617
			04	91	00079		BNEQ	SS	
			58	12	0007D		DECL	P	
			58	D7	0007F		BRB	4S	
			F6	11	00081		SUBL3	R7, P, COMMAND_DESC	
			57	C3	00083	58:	MOVCS	#0, (SP), #0, #196, ZERO_AREA	
		6A	00	2C	00087				0619
00C4 BF	6E	24	AA		0008E				0621
	0108 CA	00000000	6A	7D	00090		MOVQ	COMMAND_DESC, TPARSE_BLOCK+8	0622
		00000000	EF	9F	00095		PUSHAB	KEY_TABLE	0624
		0100	EF	9F	00098		PUSHAB	STATE_TABLE	
	00000000G	00	CA	9F	000A1		PUSHAB	TPARSE_BLOCK	
		59	03	FB	000A5		CALLS	#3, LIB\$TPARSE	
		35	50	DD	000AC		MOVL	R0, STATUS	
	00000000G	8F	59	E8	000AF		BLBS	STATUS, 78	0625
			59	D1	000B2		CMPL	STATUS, #LIB\$_SYNTAXERR	0628
			07	12	000B9		BNEQ	68	
		59 00450030	8F	DD	000B8		MOVL	#4522032, STATUS	0629
		50 0110	CA	DD	000C2	68:	MOVL	TPARSE_BLOCK+16, R0	0637
		010C DA40	9F	000C7		PUSHAB	@TPARSE_BLOCK+12[R0]		
	7E 0108 CA	0114	50	C3	000CC		SUBL3	R0, TPARSE_BLOCK+8, -(SP)	
			50	DD	000D2		PUSHL	TPARSE_BLOCK+20	
			50	DD	000D6		PUSHL	R0	
	7E 0114 CA	04	AA	DD	000D8		PUSHL	COMMAND_DESC+4	
			AA	C3	000DB		SUBL3	COMMAND_DESC+4, TPARSE_BLOCK+20, -(SP)	
			59	DD	000E2		PUSHL	STATUS	
	03 24 AA	04	07	FB	000E4		CALLS	#7, LIB\$SIGNAL	
			01	E0	000E7	78:	BB\$	#1, CLEANUP_FLAGS, 88	0640
		50	FF48	31	000EC		BRW	18	
			01	DD	000EF	88:	MOVL	#1, R0	0646
			04	000F2			RET		
			0000	000F3		98:	.WORD	Save nothing	0374
			7E	04	000F5		CLRL	-(SP)	
			SE	DD	000F7		PUSHL	SP	
	00000000V	7E EF	04	AC	000F9		MOVG	4(AP), -(SP)	
			03	FB	000FD		CALLS	#3, MAIN_HANDLER	
			04	00104			RET		

: Routine Size: 261 bytes, Routine Base: \$CODE\$ + 0000

```

653 0647 1 | Minor action routines to help out with parsing
654 0648 1 |
655 0649 1 |
656 0650 1 |
657 0651 1 |
658 0652 1 | Give invalid command status
659 0653 1 |
660 0654 1 |
661 0655 1 ROUTINE INV_COMMAND =
662 0656 1 |
663 0657 2 BEGIN
664 0658 2 TPARSE_ARGS;
665 0659 2 |
666 P 0660 2 ERR_EXIT ((IF .TPARSE_BLOCK[TPASV_AMBIG]
667 P 0661 2 THEN DSKQS_AMB_CMD
668 P 0662 2 ELSE DSKQS_INV_CMD)
669 P 0663 2 .TPARSE_BLOCK[TPASL_TOKENPTR] = .COMMAND_DESC[1],
670 P 0664 2 .COMMAND_DESC[1],
671 P 0665 2 .TPARSE_BLOCK[TPASL_TOKENCNT],
672 P 0666 2 .TPARSE_BLOCK[TPASL_TOKENPTR],
673 P 0667 2 .TPARSE_BLOCK[TPASL_STRINGCNT],
674 P 0668 2 .TPARSE_BLOCK[TPASL_STRINGPTR]
675 P 0669 2 )
676 0670 1 END;

```

0004 00000 INV_COMMAND:

				.WORD	Save R2	0655
			S2 00000000	EF 9E 00002	COMMAND_DESC+4, R2	
			7E 08	AC 7D 00009	8(TPARSE_BLOCK), -(SP)	0669
			7E 10	AC 7D 0000D	16(TPARSE_BLOCK), -(SP)	
				62 DD 00011	PUSHL	
				62 C3 00013	SUBL	
				06 AC E9 00018	COMMAND_DESC+4	
				00450010 BF DD 0001C	20(TPARSE_BLOCK), -(SP)	
				06 11 00022	BLBC	
				00450008 8F DD 00024	#4522000	
				18: 07 FB 0002A	PUSHL	
				28: 04 00031	BRB	
					CALLS	
					#7. LIB\$STOP	
					RET	

; Routine Size: 50 bytes, Routine Base: \$CODES + 0105

```

677 0671 1 |
678 0672 1 |
679 0673 1 | Give invalid switch status
680 0674 1 |
681 0675 1 |
682 0676 1 ROUTINE INV_SWITCH =
683 0677 1 |
684 0678 2 BEGIN
685 0679 2 TPARSE_ARGS;
686 0680 2 |
687 P 0681 2 ERR_EXIT ((IF .TPARSE_BLOCK[TPASV_AMBIG]

```

```
:
: 688      P 0682 2      THEN DSKQS_AMB_QUAL
: 689      P 0683 2      ELSE DSKQS_INV_QUAL),
: 690      P 0684 2      .TPARSE_BLOCK[TPASL_TOKENPTR] = .COMMAND_DESC[1],
: 691      P 0685 2      .COMMAND_DESC[1],
: 692      P 0686 2      .TPARSE_BLOCK[TPASL_TOKENCNT].
: 693      P 0687 2      .TPARSE_BLOCK[TPASL_TOKENPTR]
: 694      P 0688 2      .TPARSE_BLOCK[TPASL_STRINGCNT].
: 695      P 0689 2      .TPARSE_BLOCK[TPASL_STRINGPTR]
: 696      0690 2
: 697      0691 1 END:
```

0004 00000 INV_SWITCH:					
			.WORD	Save R2	: 0676
		52 00000000'	EF 9E 00002	MOVAB	COMMAND DESC+4, R2
		7E 08	AC 7D 00009	MOVQ	8(TPARSE_BLOCK), -(SP)
		7E 10	AC 7D 0000D	MOVQ	16(TPARSE_BLOCK), -(SP)
			62 DD 00011	PUSHL	COMMAND DESC+4
			62 C3 00013	SUBL	COMMAND DESC+4, 20(TPARSE_BLOCK), -(SP)
	7E	14 AC	AC E9 00018	BLBC	6(TPARSE_BLOCK), 1\$
		08 06	00450020	PUSHL	#4522016
			BF DD 0001C	BRB	2\$
			06 11 00022	PUSHL	#4522008
			00450018	1\$: CALLS	#7, LIB\$STOP
			BF DD 00024	2\$: RET	
		00000000G 00	07 FB 0002A		
			04 00031		

: Routine Size: 50 bytes, Routine Base: \$CODES + 0137

```
:
: 698      0692 1
: 699      0693 1
: 700      0694 1 ! Save the HELP key descriptor in the key descriptor vector.
: 701      0695 1
: 702      0696 1
: 703      0697 1 ROUTINE SAVE_KEY =
: 704      0698 1
: 705      0699 2 BEGIN
: 706      0700 2
: 707      0701 3 IF .KEY_INDEX LEQ (MAX_KEYS - 2)          ! check for too many keys
: 708      0702 2 THEN
: 709      0703 3 BEGIN
: 710      0704 3 KEY_VECTOR[.KEY_INDEX] = .TPARSE_BLOCK[TPASL_TOKENCNT];
: 711      0705 3 KEY_VECTOR[.KEY_INDEX+1] = .TPARSE_BLOCK[TPASL_TOKENPTR];
: 712      0706 3 KEY_INDEX = .KEY_INDEX+2;                  ! Increment KEY_INDEX
: 713      0707 2 END;
: 714      0708 2
: 715      0709 1 END;
```

0004 00000 SAVE_KEY:					
			.WORD	Save R2	: 0697

52 00000000'	EF 9E 00002	MOVAB KEY_INDEX, R2	
50	62 D0 00009	MOVL KEY_INDEX, R0	0701
OC	50 D1 0000C	CMPL R0, #12	
C8 A240	0F 14 0000F	BGTR 1S	
CC A240	30 A2 D0 00011	MOVL TPARSE_BLOCK+16, KEY_VECTOR[R0]	0704
62	A2 D0 00017	MOVL TPARSE_BLOCK+20, KEY_VECTOR+4[R0]	0705
50	02 C0 0001D	ADDL2 #2, KEY_INDEX	0706
	01 D0 00020	MOVL #1, R0	0709
	04 00023	RET	

: Routine Size: 36 bytes, Routine Base: \$CODES + 0169

```
: 717      0710 1 GLOBAL ROUTINE USE_DEFAULT : NOVALUE =
: 718      0711 1
: 719      0712 1 ++
: 720      0713 1
: 721      0714 1 Functional Description:
: 722      0715 1
: 723      0716 1 This routine causes a USE SYSSDISK: command to be executed, to
: 724      0717 1 set up the channel to the default disk. If it fails, no error
: 725      0718 1 messages are output and the channel is simply left unassigned.
: 726      0719 1
: 727      0720 1 Calling Sequence:
: 728      0721 1 standard
: 729      0722 1
: 730      0723 1 Input Parameters:
: 731      0724 1 none
: 732      0725 1
: 733      0726 1 Implicit Inputs:
: 734      0727 1 none
: 735      0728 1
: 736      0729 1 Output Parameters:
: 737      0730 1 none
: 738      0731 1
: 739      0732 1 Implicit Outputs:
: 740      0733 1 none
: 741      0734 1
: 742      0735 1 Routines Called:
: 743      0736 1 none
: 744      0737 1
: 745      0738 1 Routine Value:
: 746      0739 1 none
: 747      0740 1
: 748      0741 1 Signals:
: 749      0742 1 none
: 750      0743 1
: 751      0744 1 Side Effects:
: 752      0745 1 none
: 753      0746 1
: 754      0747 1 --
: 755      0748 1
: 756      0749 2 BEGIN
: 757      0750 2
: 758      0751 2 BUILTIN
: 759      0752 2 CALLG; : linkage to action routines is CALLG
: 760      0753 2
: 761      0754 2
: 762      0755 2 : Enable the local condition handler to swallow error signals. Then plug
: 763      0756 2 the TPARSE control block and call the USE action routine.
: 764      0757 2
: 765      0758 2
: 766      0759 2 ENABLE DEF_HANDLER;
: 767      0760 2
: 768      0761 2 TPARSE_BLOCK[TPASL_TOKENCNT] = %CHARCOUNT ('SYSSDISK:');
: 769      0762 2 TPARSE_BLOCK[TPASL_TOKENPTR] = UPLIT BYTE ('SYSSDISK:');
: 770      0763 2 CALLG TPARSE_BLOCK, ACT_USE;
: 771      0764 2
: 772      0765 1 END; ! end of routine USE_DEFAULT
```

3A 4B 53 49 44 24 53 59 53 00110 P.AAF: .ASCII \SYS\$DISK:\PSECT SPLITS,NOWRT,NOEXE,2

				.PSECT	SCODES,NOWRT,2
52	00000000	'	0004 00000	.ENTRY	USE DEFAULT, Save R2
6D	0015	'	EF 9E 00002	MOVAB	TPARSE_BLOCK+16, R2
62			CF DE 00009	MOVAL	1S, (FP)
04	A2 00000000	'	09 D0 0000E	MOVL	#9, TPARSE_BLOCK+16
00000000V	EF	FO	A2 FA 00011	MOVAB	P_AAF, TPARSE_BLOCK+20
			A2 FA 00019	CALLG	TPARSE_BLOCK,_ACT_USE
			04 00021	RET	
			0000 00022	.WORD	Save nothing
			7E D4 00024	CLRL	-(SP)
			5E DD 00026	PUSHL	SP
00000000V	7E	04	AC 7D 00028	MOVO	4(AP), -(SP)
			03 FB 0002C	CALLS	#3, DEF_HANDLER
			04 00033	RET	

; Routine Size: 52 bytes, Routine Base: SCODES + 018D

```

774 0766 1 GLOBAL ROUTINE DEF_HANDLER (SIGNAL, MECHANISM) : NOVALUE =
775 0767 1
776 0768 1 ++
777 0769 1
778 0770 1 Functional Description:
779 0771 1
780 0772 1 This routine is the condition handler for the preceding routine.
781 0773 1 It simply unwinds the stack on any signal.
782 0774 1
783 0775 1 Calling Sequence:
784 0776 1 standard
785 0777 1
786 0778 1 Input Parameters:
787 0779 1 none
788 0780 1
789 0781 1 Implicit Inputs:
790 0782 1 none
791 0783 1
792 0784 1 Output Parameters:
793 0785 1 none
794 0786 1
795 0787 1 Implicit Outputs:
796 0788 1 none
797 0789 1
798 0790 1 Routines Called:
799 0791 1 none
800 0792 1
801 0793 1 Routine Value:
802 0794 1 none
803 0795 1
804 0796 1 Signals:
805 0797 1 none
806 0798 1
807 0799 1 Side Effects:
808 0800 1 none
809 0801 1
810 0802 1 --
811 0803 1
812 0804 2 BEGIN
813 0805 2
814 0806 2 MAP
815 0807 2 SIGNAL : REF BBLOCK, ! signal vector
816 0808 2 MECHANISM : REF BBLOCK; ! mechanism vector
817 0809 2
818 0810 2
819 0811 2 $UNWIND ();
820 0812 2
821 0813 1 END; ! end of routine DEF_HANDLER

```

.EXTRN SY\$UNWIND

00000000G 00	0000 00000 7E 7C 00002 02 FB 00004 04 00008	.ENTRY DEF_HANDLER. Save nothing CLRQ -(SP) CALLS #2, SY\$UNWIND RET
--------------	--	---

: 0766
: 0811
: 0813

DISKQUOTA
V04-000

8 12
15-Sep-1984 23:38:38
14-Sep-1984 12:19:46 VAX-11 Bliss-32 v4.0-742
[DISK0.SRC]DISKQUOTA.B32;1

Page 38
(9)

: Routine Size: 12 bytes, Routine Base: \$CODE\$ + 01C1

```
823 0814 1 GLOBAL ROUTINE ACT_USE =
824 0815 1
825 0816 1 ++
826 0817 1
827 0818 1 Functional Description:
828 0819 1
829 0820 1 This action routine processes the USE command. It assigns a channel
830 0821 1 to the specified device string.
831 0822 1 Calling Sequence:
832 0823 1 standard
833 0824 1
834 0825 1 Input Parameters:
835 0826 1 none
836 0827 1
837 0828 1 Implicit Inputs:
838 0829 1 none
839 0830 1
840 0831 1 Output Parameters:
841 0832 1 none
842 0833 1
843 0834 1 Implicit Outputs:
844 0835 1 none
845 0836 1
846 0837 1 Routines Called:
847 0838 1 none
848 0839 1
849 0840 1 Routine Value:
850 0841 1 none
851 0842 1
852 0843 1 Signals:
853 0844 1 none
854 0845 1
855 0846 1 Side Effects:
856 0847 1 none
857 0848 1
858 0849 1
859 0850 1 !--
860 0851 1
861 0852 2 BEGIN
862 0853 2
863 0854 2 LITERAL
864 0855 2     BUFFER_LEN      = 64;           ! string buffer length
865 0856 2
866 0857 2 LOCAL
867 0858 2     P;                      ! general string pointer
868 0859 2     STATUS;                 ! general status value
869 0860 2     NAME_DESC : VECTOR [2];   ! descriptor of logical name to translate
870 0861 2     RESULT : VECTOR [2];    ! descriptor of translated name
871 0862 2     STRING_BUFFER : VECTOR [BUFFER_LEN, BYTE]; ! string buffer (obviously)
872 0863 2
873 0864 2 TPARSE_ARGS;               ! declare TPARSE argument list
874 0865 2
875 0866 2
876 0867 2 ! Get the device name string and attempt to do logical name translation.
877 0868 2 We iterate on logical name translation until the service returns SSS_NOTRAN.
878 0869 2 Perform device name extraction by using only the part of the logical name to
879 0870 2 the left of the colon (if any), also checking for node names.
```

```

880 0871 2 !
881 0872 2
882 0873 2 IF .CHANNEL NEQ 0
883 0874 2 THEN $DASSGN (CHAN = .CHANNEL);
884 0875 2 CHANNEL = 0;
885 0876 2
886 0877 2 RESULT[0] = BUFFER_LEN;
887 0878 2 RESULT[1] = STRING_BUFFER;
888 0879 2 NAME_DESC[0] = .TPARSE_BLOCK[TPASL_TOKENCNT];           ! get initial logical name
889 0880 2 NAME_DESC[1] = STRING_BUFFER;
890 0881 2 CHSCOPY (.TPARSE_BLOCK[TPASL_TOKENCNT], .TPARSE_BLOCK[TPASL_TOKENPTR], 0, .RESULT[0], .RESULT[1]);
891 0882 2
892 0883 3 IF BEGIN
893 0884 3   DECR N FROM 10 TO 1 DO
894 0885 4     BEGIN
895 0886 4       P = CHSFIND CH (.NAME_DESC[0], .NAME_DESC[1], ':');
896 0887 4       IF NOT CHSFFAIL (.P)
897 0888 4         THEN
898 0889 5           BEGIN
899 0890 5             IF .P - .NAME_DESC[1] LSSU .NAME_DESC[0] - 1
900 0891 5               AND .(P)<0,18> EQL '::'
901 0892 5               THEN ERR_EXIT (DSKQS_NONLOCAL);
902 0893 5             NAME_DESC[0] = .P - .NAME_DESC[1];
903 0894 4           END;
904 0895 4
905 0896 4       IF CHSRCHAR (.NAME_DESC[1]) EQL '_'
906 0897 4         THEN EXITLOOP 0;
907 0898 4
908 P 0899 4       STATUS = STRNLOG (LOGNAM = NAME_DESC[0],
909 P 0900 4                     RSLLEN = NAME_DESC[0],
910 P 0901 4                     RSLBUF = RESULT[0]);
911 0902 4       IF .STATUS EQL SSS_NOTRAN THEN EXITLOOP 0;
912 0903 4       IF NOT .STATUS THEN ERR_EXIT (.STATUS);
913 0904 4     END
914 0905 3
915 0906 2 THEN ERR_EXIT (DSKQS_NOTRAN);
916 0907 2
917 0908 2 RESULT[0] = .NAME_DESC[0];
918 0909 2
919 0910 2 ! Now assign a channel to the device name.
920 0911 2
921 0912 2
922 0913 2 STATUS = SASSIGN (DEVNAM = RESULT[0], CHAN = CHANNEL);
923 0914 2 IF NOT .STATUS
924 0915 2 THEN ERR_EXIT (.STATUS);
925 0916 2
926 0917 2 1
927 0918 2
928 0919 1 END;                                ! end of routine ACT_USE

```

.EXTRN SYSSDASSGN, SYSSTRNLOG
.EXTRN SYSSASSIGN

.ENTRY ACT USE, Save R2,R3,R4,R5,R6,R7
MOVAB CHANNEL, R7

57 00000000' EF 00FC 00000
00000002

: 0814

			56 00000000G	00 9E 00009	MOVAB LIB\$STOP, R6	
			5E B0	A0 9E 00010	MOVAB -80(SP), SP	0873
			50	67 3C 00014	MOVZWL CHANNEL, R0	
				09 13 00017	BEQL 1S	
				50 DD 00019	PUSHL R0	
			00000000G 00	01 FB 0001B	CALLS #1, SYSSDASSGN	0874
				67 B4 00022	CLRW CHANNEL	0875
			40 AE	8F 9A 00024	MOVZBL #64 RESULT	0877
			44 AE	6E 9E 00029	MOVAB STRING BUFFER, RESULT+4	0878
			48 AE	AC D0 0002D	MOVL 16(TPARSE_BLOCK), NAME DESC	0879
			4C AE	6E 9E 00032	MOVAB STRING BUFFER, NAME DESC+4	0880
40 AE	00	14 BC	10 44	AC 2C 00036	MOVCS 16(TPARSE_BLOCK), #20(TPARSE_BLOCK), #0, -	0881
				BE 0003E	RESULT, @RESULT+4	
				0A D0 00040	MOVL #10, N	0884
			4C BE	3A 00043	LOC C #58, NAME_DESC, @NAME_DESC+4	0886
				02 12 00049	BNEQ 3S	
				51 D4 0004B	CLRL R1	
				51 D0 0004D	MOVL R1, P	
			52 50	23 13 00050	BEQL 5S	
				AE C3 00052	SUBL3 NAME_DESC+4, P, R2	0887
			48 AE	01 C3 00057	SUBL3 #1, NAME_DESC, R0	0890
			50	52 D1 0005C	CMPL R2, R0	
				10 1E 0005F	BGEQU 4S	
			3A3A	65 B1 00061	CMPW (P), #14906	0891
				09 12 00066	BNEQ 4S	
				8F DD 00068	PUSHL #4522040	0892
			48 5F	01 FB 0006E	CALLS #1, LIB\$STOP	
				52 D0 00071	MOVL R2, NAME DESC	0893
				91 00075	CMPB @NAME_DESC+4, #95	0896
				34 13 0007A	BEQL 7S	
				7E 7C 0007C	CLRQ -(SP)	0901
				7E D4 0007E	CLRL -(SP)	
				4C AE 9F 00080	PUSHAB RESULT	
				58 AE 9F 00083	PUSHAB NAME_DESC	
				5C AE 9F 00086	PUSHAB NAME_DESC	
			00000000G 00	06 FB 00089	CALLS #6, SYSSTRNLOG	
			53	50 D0 00090	MOVL R0, STATUS	
			00000629 8F	53 D1 00093	CMPL STATUS, #1577	0902
				14 13 0009A	BEQL 7S	
				53 E8 0009C	BLBS STATUS, 6S	0903
				53 DD 0009F	PUSHL STATUS	
				05	CALLS #1, LIB\$STOP	
			66 9C	01 FB 000A1	N, 2S	0884
				54 F5 000A4	SOBGTR #4522048	0906
				8F DD 000A7	PUSHL #1, LIB\$STOP	
			60 AE	01 FB 000AD	CALLS NAME DESC, RESULT	0908
				7E 7C 000B5	MOVL -(SP)	0913
				57 DD 000B7	CLRQ R7	
			00000000G 00	AE 9F 000B9	PUSHAB RESULT	
			53	04 FB 000BC	CALLS #4, SYSSASSIGN	
			05	50 D0 000C3	MOVL R0, STATUS	
			66 50	23 E8 000C6	BLBS STATUS, 8S	0914
				23 DD 000C9	PUSHL STATUS	0915
				01 FB 000CB	CALLS #1, LIB\$STOP	
				01 D0 000CE	MOVL #1, R0	
				04 000D1	RET	0919

DISKQUOTA
V04-000

5 12
15-Sep-1984 23:38:38
14-Sep-1984 12:19:46

VAX-11 Bliss-32 V4.0-742
[DISKO.SRC]DISKQUOTA.B32;1

Page 42
(10)

: Routine Size: 210 bytes. Routine Base: \$CODES + 01CD

DI
VO

```
930      0920 1 GLOBAL ROUTINE ACT_CREATE =
931      0921 1
932      0922 1   ++
933      0923 1
934      0924 1   Functional Description:
935      0925 1
936      0926 1     This action routine implements the CREATE command. It creates the
937      0927 1     disk quota file and activates it.
938      0928 1
939      0929 1   Calling Sequence:
940      0930 1     Standard
941      0931 1
942      0932 1   Input Parameters:
943      0933 1     none
944      0934 1
945      0935 1   Implicit Inputs:
946      0936 1     none
947      0937 1
948      0938 1   Output Parameters:
949      0939 1     none
950      0940 1
951      0941 1   Implicit Outputs:
952      0942 1     none
953      0943 1
954      0944 1   Routines Called:
955      0945 1     none
956      0946 1
957      0947 1   Routine Value:
958      0948 1     none
959      0949 1
960      0950 1   Signals:
961      0951 1     none
962      0952 1
963      0953 1   Side Effects:
964      0954 1     none
965      0955 1
966      0956 1   --
967      0957 1
968      0958 2 BEGIN
969
970      0960 2 BIND      FILE_DATA      = UPLIT (1, 0, 0, 1000, 100, REP 123 OF (0));
971      0961 2
972      0962 2 PSECT      PLIT        = SOWNS;
973      0963 2
974      0964 2
975      0965 2
976      0966 2 BIND      CREATE_ATTRIB = UPLIT (WORD (FATSC_LENGTH, ATRSC_RECATTR),
977      0967 2           UPLIT (BYTE TFATSC_FIXED, 0), WORD (DQFSC_LENGTH),
978      0968 2           1^16, 2^16, WORD (0, 0, DQFSC_LENGTH, 0)),
979      0969 2           0);
980      0970 2
981      0971 2 PSECT      PLIT        = SPLITS;
982      0972 2
983      0973 2
984      0974 2 LOCAL      STATUS;    ! general status value
985      0975 2
986      0976 2
```

```
987 0977 2
988 0978 2
989 0979 2
990 0980 2
991 0981 2
992 0982 2
993 0983 2
994 0984 2
995 0985 2
996 0986 2
997 0987 2
998 0988 2
999 0989 2
1000 0990 2
1001 0991 2
1002 0992 2
1003 0993 2
1004 0994 2
1005 0995 2
1006 0996 2
1007 P 0997 2
1008 P 0998 2
1009 P 0999 2
1010 P 1000 2
1011 P 1001 2
1012 P 1002 2
1013 1003 2
1014 1004 2
1015 1005 2
1016 1006 2
1017 1007 2
1018 1008 2
1019 1009 2
1020 1010 2
1021 P 1011 2
1022 P 1012 2
1023 P 1013 2
1024 P 1014 2
1025 P 1015 2
1026 P 1016 2
1027 1017 2
1028 1018 2
1029 1019 2
1030 1020 2
1031 1021 2
1032 P 1022 2
1033 P 1023 2
1034 P 1024 2
1035 1025 2
1036 1026 2
1037 1027 2
1038 1028 2
1039 1029 2
1040 1030 2
1041 1031 2
1042 1032 2
1043 1033 2

; Verify that a channel is open.

IF .CHANNEL EQL 0
THEN ERR_EXIT (DSKOS_NODEVICE);

; Create the quota file.

QUOTA_FIB[FIBSW_DID_NUM] = FIDSC_MFD;
QUOTA_FIB[FIBSW_DID_SEQ] = FIDSC_MFD;
QUOTA_FIB[FIBSW_DID_RVN] = 1;
QUOTA_FIB[FIBSL_ACCTL] = FIBSM_WRITE OR FIBSM_NOREAD;
QUOTA_FIB[FIBSW_EXCTL] = FIBSM_EXTEND OR FIBSM_ALCON OR FIBSM_FILCON;
QUOTA_FIB[FIBSL_EXSZ] = 1;
QUOTA_FIB[FIBSB_ALALIGN] = FIBSC_LBN;
QUOTA_FIB[FIBSW_LOC_RVN] = 1;

STATUS = SQIOW (CHAN = .CHANNEL,
                 FUNC = IOS_CREATE OR IOSM_CREATE OR IOSM_ACCESS,
                 IOSB = IO_STATUS,
                 P1 = OFIB_DESC,
                 P2 = OFILE_NAME,
                 P5 = CREATE_ATTRIB
               );
IF .STATUS THEN STATUS = .IO_STATUS[0];
IF NOT .STATUS
THEN ERR_EXIT (DSKOS_CREATERR, .STATUS);

; Write the initial data block and close the file.

STATUS = SQIOW (CHAN = .CHANNEL,
                 FUNC = IOS_WRITEVBLK,
                 IOSB = IO_STATUS,
                 P1 = OFILE_DATA,
                 P2 = 512,
                 P3 = 1
               );
IF .STATUS THEN STATUS = .IO_STATUS[0];
IF NOT .STATUS
THEN ERR_EXIT (DSKOS_INITERR, .STATUS);

STATUS = SQIOW (CHAN = .CHANNEL,
                 FUNC = IOS_DEACCESS,
                 IOSB = IO_STATUS
               );
IF .STATUS THEN STATUS = .IO_STATUS[0];
IF NOT .STATUS
THEN ERR_EXIT (DSKOS_CLOSERR, .STATUS);

; Now activate the quota file.

QUOTA_FIB[FIBSW_DID_NUM] = 0;
```

```

1044 1034 2 QUOTA_FIB[FIBSW_DID_SEQ] = 0;
1045 1035 2 QUOTA_FIB[FIBSW_DID_RVN] = 0;
1046 1036 2 QUOTA_FIB[FIBSW_CTRL FUNC] = FIBSC_ENA_QUOTA;
1047 1037 2 QUOTA_FIB[FIBSL_CTRL VAL] = 0;
1048 P 1038 2 STATUS = SQIOW TCHAN = .CHANNEL,
1049 P 1039 2 FUNC = IOS ACP CONTROL,
1050 P 1040 2 IOSB = IO STATUS,
1051 P 1041 2 P1 = QFIB_DESC
1052 1042 2 );
1053 1043 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
1054 1044 2 IF NOT .STATUS
1055 1045 2 THEN ERR_EXIT(DSKOS_ACTERR, .STATUS);
1056 1046 2
1057 1047 2
1058 1048 1 END:

```

! end of routine ACT_CREATE

				.PSECT SPLITS,NOWRT,NOEXE,2
00000064	000003E8	00000000	00000000	00119
				0011C P.AAG:
				0000000# 00130
				.BLKB 3
				.LONG 1 0 0, 1000, 100
				.LONG 0{12\$}
				.PSECT SOWNS,NOEXE,2
				00 01 0024C P.AAI:
				0020 0024E
				.BYTE 1 0
				00020000 00010000 00250
				.WORD 32
				0000 0020 0000 0000 00258
				.LONG 65536, 131072
				0004 0020 00260 P.AAH:
				.WORD 0 0 {32. 0}
				00000000 00264
				.WORD 32. 4
				00000000 00268
				.ADDRESS P.AAI
				.LONG 0
				QFILE DATA=
				CREATE_ATTRIB=
				P.AAG
				.EXTRN SYSSQIOW
				P.AAH
				.PSECT SCODES,NOWRT,2
				.ENTRY ACT CREATE, Save R2,R3,R4,R5
				MOVAB SYSSQIOW, R5
				MOVAB LIBSTOP, R4
				MOVAB IO STATUS, R3
				TSTW CHANNEL
				BNEQ 18
				PUSHL #4522056
				CALLS #1 LIBSTOP
				MOVL #262148, QUOTA_FIB+10
				MOVW #1 QUOTA_FIB+T4
				MOVZUL #1280, QUOTA_FIB
				MOVZBW #133, QUOTA_FIB+22
				MOVL #1, QUOTA_FIB+24
				MOVB #2, QUOTA_FIB+33
				MOVW #1 QUOTA_FIB+38
				CLRL -(SP)
				PUSHAB CREATE_ATTRIB
				0920
				0982
				0983
				0988
				0990
				0991
				0992
				0993
				0994
				0995
				1003

B 13
15-Sep-1984 23:38:38 VAX-11 Bliss-32 v4.0-742
14-Sep-1984 12:19:46 [DISK0.SRC]DISKQUOTA.B32;1

Page 46
(11)

DIS
VO4

			7E	7C	00055		CLRQ	-(SP)
			0240	C3	9F	00057	PUSHAB	QFILE NAME
			0208	C3	9F	0005B	PUSHAB	QFIB DESC
				7E	7C	0005F	CLRQ	-(SP)
				53	DD	00061	PUSHL	R3
				8F	9A	00063	MOVZBL	#243, -(SP)
				A3	3C	00067	MOVZWL	CHANNEL, -(SP)
				7E	D4	0006B	CLRL	-(SP)
				0C	FB	0006D	CALLS	#12, SYSSQIOW
				50	DO	00070	MOVL	RO, STATUS
				52	E9	00073	BLBC	STÁTUS, 28
				63	3C	00076	MOVZWL	IO STATUS STATUS
				552	E8	00079	BLBS	STATUS, 38
				52	DD	0007C	PUSHL	STATUS
				8F	DD	0007E	PUSHL	#4522064
				02	FB	00084	CALLS	#2, LIB\$STOP
				7E	7C	00087	CLRQ	-(SP)
				01	7D	00089	MOVO	#1, -(SP)
				8F	3C	0008C	MOVZWL	#512, -(SP)
				EF	9F	00091	PUSHAB	QFILE DATA
				7E	7C	00097	CLRQ	-(SP)
				53	DD	00099	PUSHL	R3
				30	DD	0009B	PUSHL	#48
				A3	3C	0009D	MOVZWL	CHANNEL, -(SP)
				7E	D4	000A1	CLRL	-(SP)
				0C	FB	000A3	CALLS	#12, SYSSQIOW
				50	DO	000A6	MOVL	RO, STATUS
				52	E9	000A9	BLBC	STÁTUS, 48
				63	3C	000AC	MOVZWL	IO STATUS STATUS
				52	E8	000AF	BLBS	STATUS, 58
				52	DD	000B2	PUSHL	STATUS
				8F	DD	000B4	PUSHL	#4522072
				02	FB	000BA	CALLS	#2, LIB\$STOP
				7E	7C	000BD	CLRQ	-(SP)
				7E	7C	000BF	CLRQ	-(SP)
				7E	7C	000C1	CLRQ	-(SP)
				7E	7C	000C3	CLRQ	-(SP)
				53	DD	000C5	PUSHL	R3
				34	DD	000C7	PUSHL	#52
				A3	3C	000C9	MOVZWL	CHANNEL, -(SP)
				7E	D4	000CD	CLRL	-(SP)
				0C	FB	000CF	CALLS	#12, SYSSQIOW
				50	DO	000D2	MOVL	RO, STATUS
				52	E9	000D5	BLBC	STÁTUS, 68
				63	3C	000D8	MOVZWL	IO STATUS STATUS
				52	E8	000DB	BLBS	STATUS, 78
				52	DD	000DE	PUSHL	STATUS
				8F	DD	000E0	PUSHL	#4522080
				02	FB	000E6	CALLS	#2, LIB\$STOP
				C3	D4	000E9	CLRL	QUOTA_FIB+10
				C3	B4	000ED	CLRW	QUOTA_FIB+14
				09	B0	000F1	MOVW	#9, QQUOTA_FIB+22
				C3	D4	000F6	CLRL	QUOTA_FIB+24
				7E	7C	000FA	CLRQ	-(SP)
				7E	7C	000FC	CLRU	-(SP)
				7E	D4	000FE	CLRL	-(SP)
				0208	C3	9F	PUSHAB	QFIB DESC

		7E	7C 00104	CLRQ -(SP)	
		53	DD 00106	PUSHL R3	
		58	DD 00108	PUSHL #56	
	7E	A3	3C 0010A	MOVZWL CHANNEL, -(SP)	
65		7E	D4 0010E	CLRL -(SP)	
52		0C	FB 00110	CALLS #12, SYSSQIOW	
06		50	DD 00113	MOVL R0, STATUS	
52		52	E9 00116	BLBC STATUS, 88	1043
08		63	3C 00119	MOVZWL IO STATUS STATUS	
	00450068	52	E8 0011C	BLBS STATUS, 98	
		52	DD 0011F	PUSHL STATUS	1044
64		8F	DD 00121	PUSHL #4522088	
50		02	FB 00127	CALLS #2, LIB\$STOP	
		01	DD 0012A	MOVL #1, R0	1045
		04	0012D	RET	

: Routine Size: 302 bytes, Routine Base: \$CODE\$ + 029F

20
20

72

55
3321
314C
6F72
66

```
; 1060
; 1061
; 1062
; 1063
; 1064
; 1065
; 1066
; 1067
; 1068
; 1069
; 1070
; 1071
; 1072
; 1073
; 1074
; 1075
; 1076
; 1077
; 1078
; 1079
; 1080
; 1081
; 1082
; 1083
; 1084
; 1085
; 1086
; 1087
; 1088
; 1089
; 1090
; 1091
; 1092
; 1093
; 1094
; 1095
; 1096
; 1097
; 1098
; 1099
; 1100
; 1101
; 1102
; 1103
; 1104
; 1105
; 1106
; 1107
; 1108
; 1109
; 1110
; 1111
; 1112
; 1113
; 1114
; 1115
; 1116

1049 1 GLOBAL ROUTINE ACT_ENABLE =
1050 1 /**
1051 1 : Functional Description:
1052 1 : This action routine implements the ENABLE command. It enables the
1053 1 : disk quota file.
1054 1 :
1055 1 :
1056 1 :
1057 1 :
1058 1 : Calling Sequence:
1059 1 : standard
1060 1 :
1061 1 : Input Parameters:
1062 1 : none
1063 1 :
1064 1 : Implicit Inputs:
1065 1 : none
1066 1 :
1067 1 : Output Parameters:
1068 1 : none
1069 1 :
1070 1 : Implicit Outputs:
1071 1 : none
1072 1 :
1073 1 : Routines Called:
1074 1 : none
1075 1 :
1076 1 : Routine Value:
1077 1 : none
1078 1 :
1079 1 : Signals:
1080 1 : none
1081 1 :
1082 1 : Side Effects:
1083 1 : none
1084 1 :
1085 1 : ---
1086 1 :
1087 1 : BEGIN
1088 2 :
1089 2 :
1090 2 LOCAL STATUS; ! general status value
1091 2 :
1092 2 :
1093 2 :
1094 2 : Verify that a channel is open.
1095 2 :
1096 2 :
1097 2 IF .CHANNEL EQL 0
1098 2 THEN ERR_EXIT (DSKQ$_NODEVICE);
1099 2 :
1100 2 : Now activate the quota file.
1101 2 :
1102 2 :
1103 2 QUOTA_FIB[FIBSW_DID_NUM] = FIDSC_MFD;
1104 2 QUOTA_FIB[FIBSW_DID_SEQ] = FIDSC_MFD;
1105 2 QUOTA_FIB[FIBSW_DID_RVN] = 1;
```

```

1117      1106 2 QUOTA_FIB[FIBSW.CNTRLFUNC] = FIBSC_ENA_QUOTA;
1118      1107 2 STATUS = $QIOW TCHAN = .CHANNEL,
1119          1108 2 FUNC = IOS_ACPCONTROL,
1120          1109 2 IOSB = IO_STATUS,
1121          1110 2 P1 = QFIB_DESC,
1122          1111 2 P2 = QFILE_NAME
1123          1112 2 );
1124          1113 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
1125          1114 2 IF NOT .STATUS
1126          1115 2 THEN ERR_EXIT(DSKQS_ACTERR, .STATUS);
1127          1116 2
1128          1117 2 1
1129          1118 1 END:

```

! end of routine ACT_ENABLE

53	00000000G	0000C	000000	.	ENTRY	ACT ENABLE, Save R2,R3					1049
52	00000000	00	9E 00002		MOVAB	LIB\$STOP, R3					
		EF	9E 00009		MOVAB	CHANNEL, R2					
		62	B5 00010		TSTW	CHANNEL					
		09	12 00012		BNEQ	1\$					
	00450068	8F	DD 00014		PUSHL	#4522056					
63	00040004	01	FB 0001A	18:	CALLS	#1 LIB\$STOP					1098
0186	C2	8F	D0 0001D		MOVL	#262148, QUOTA_FIB+10					1103
018A	C2	01	B0 00026		MOVW	#1, QUOTA_FIB+T4					1105
0192	C2	09	B0 0002B		MOVW	#9, QUOTA_FIB+22					1106
		7E	7C 00030		CLRQ	-(SP)					1112
		7E	7C 00032		CLRQ	-(SP)					
0244	C2	9F	00034		PUSHAB	QFILE NAME					
020C	C2	9F	00038		PUSHAB	QFIB DESC					
		7E	7C 0003C		CLRQ	-(SP)					
04	A2	9F	0003E		PUSHAB	IO STATUS					
		38	DD 00041		PUSHL	#58					
7E		62	3C 00043		MOVZWL	CHANNEL, -(SP)					
00000000G	00		7E D4 00046		CLRL	-(SP)					
	07		OC FB 00048		CALLS	#12, SYSSQIOW					
	50		50 F9 0004F		BLBC	STATUS, 2\$					
	50	04	A2 3C 00052		MOVZWL	IO STATUS STATUS					1113
	00		50 E8 00056		BLBS	STATUS, 3\$					1114
			50 DD 00059	28:	PUSHL	STATUS					1115
	00450068	8F	DD 0005B		PUSHL	#4522088					
63		02	FB 00061		CALLS	#2, LIB\$STOP					
50		01	D0 00064	38:	MOVL	#1, R0					
		04	00067		RET						1118

; Routine Size: 104 bytes, Routine Base: SCODES + 03CD

```
1131    1119 1 GLOBAL ROUTINE ACT_DISABLE =
1132    1120 1
1133    1121 1 ++
1134    1122 1
1135    1123 1 Functional Description:
1136    1124 1
1137    1125 1 This action routine implements the DISABLE command. It disables the
1138    1126 1 disk quota file.
1139    1127 1
1140    1128 1 Calling Sequence:
1141    1129 1 standard
1142    1130 1
1143    1131 1 Input Parameters:
1144    1132 1 none
1145    1133 1
1146    1134 1 Implicit Inputs:
1147    1135 1 none
1148    1136 1
1149    1137 1 Output Parameters:
1150    1138 1 none
1151    1139 1
1152    1140 1 Implicit Outputs:
1153    1141 1 none
1154    1142 1
1155    1143 1 Routines Called:
1156    1144 1 none
1157    1145 1
1158    1146 1 Routine Value:
1159    1147 1 none
1160    1148 1
1161    1149 1 Signals:
1162    1150 1 none
1163    1151 1
1164    1152 1 Side Effects:
1165    1153 1 none
1166    1154 1
1167    1155 1 --
1168    1156 1
1169    1157 2 BEGIN
1170    1158 2
1171    1159 2
1172    1160 2 LOCAL STATUS; ! general status_value
1173    1161 2
1174    1162 2
1175    1163 2
1176    1164 2 ! Verify that a channel is open.
1177    1165 2 !
1178    1166 2
1179    1167 2 IF .CHANNEL EQL 0
1180    1168 2 THEN ERR_EXIT (DSKQS_NODEVICE);
1181    1169 2
1182    1170 2 ! Now deactivate the quota file.
1183    1171 2 !
1184    1172 2
1185    1173 2 QUOTA_FIB[FIB$W_CNTRLFUNC] = FIB$C_DSA_QUOTA;
1186    1174 2 QUOTA_FIB[FIB$L_CNTRLVAL] = 0;
P 1187    1175 2 STATUS = $010W TCHAN = .CHANNEL.
```

; Routine Size: 92 bytes, Routine Base: SCODES + 0435

```
1199      1186 1 GLOBAL ROUTINE ACT_ADD =
1200      1187 1
1201      1188 1 ++
1202      1189 1
1203      1190 1 Functional Description:
1204      1191 1
1205      1192 1 This action routine implements the ADD command. It adds the
1206      1193 1 specified entry to the quota file.
1207      1194 1
1208      1195 1 Calling Sequence:
1209      1196 1 standard
1210      1197 1
1211      1198 1 Input Parameters:
1212      1199 1 none
1213      1200 1
1214      1201 1 Implicit Inputs:
1215      1202 1 none
1216      1203 1
1217      1204 1 Output Parameters:
1218      1205 1 none
1219      1206 1
1220      1207 1 Implicit Outputs:
1221      1208 1 none
1222      1209 1
1223      1210 1 Routines Called:
1224      1211 1 none
1225      1212 1
1226      1213 1 Routine Value:
1227      1214 1 none
1228      1215 1
1229      1216 1 Signals:
1230      1217 1 none
1231      1218 1
1232      1219 1 Side Effects:
1233      1220 1 none
1234      1221 1
1235      1222 1 --
1236      1223 1
1237      1224 2 BEGIN
1238      1225 2
1239      1226 2
1240      1227 2 LOCAL STATUS: ! general status value
1241      1228 2
1242      1229 2
1243      1230 2
1244      1231 2 ! Verify that a channel is open.
1245      1232 2
1246      1233 2
1247      1234 2 IF .CHANNEL EQL 0
1248      1235 2 THEN ERR_EXIT (DSKQS_NODEVICE);
1249      1236 2
1250      1237 2 ! Validate the UIC to insure that there are no wildcards.
1251      1238 2
1252      1239 2
1253      1240 2 IF .UIC_VALUE<16,16> EQL UIC$K_WILD_GROUP
1254      1241 2 OR .UIC_VALUE<0,16> EQL UIC$K_WILD_MEMBER
1255      1242 2 THEN ERR_EXIT (DSKQS_INV_UIC);
```

```

1256 1243 2
1257 1244 2 | If either value is not specified, read the default record and copy its
1258 1245 2 | values into the unspecified fields.
1259 1246 2
1260 1247 2
1261 1248 2 IF NOT .UIC_FLAGS[PERM_SPEC]
1262 1249 2 OR NOT .UIC_FLAGS[OVER_SPEC]
1263 1250 2 THEN
1264 1251 2 BEGIN
1265 1252 2 QUOTA_FIB[FIBSW_CNTRLFUNC] = FIBSC_EXA_QUOTA;
1266 1253 2 QUOTA_FIB[FIBSL_CNTRLVAL] = 0;
1267 1254 2 QUOTA_FIB[FIBSL_WCC] = 0;
1268 1255 2 STATUS = $QIOW T[CHAN = .CHANNEL,
1269 1256 2 FUNC = IOS_ACPCONTROL,
1270 1257 2 IOSB = IO_STATUS,
1271 1258 2 P1 = QFIB_DESC,
1272 1259 2 P2 = DSTREC_DESC,
1273 1260 2 P4 = DSTREC_DESC
1274 1261 2 );
1275 1262 2 IF NOT .UIC_FLAGS[PERM_SPEC]
1276 1263 2 THEN PERM_VALUE = .DST_REC[DQFSL_PERMQUOTA];
1277 1264 2 IF NOT .UIC_FLAGS[OVER_SPEC]
1278 1265 2 THEN OVER_VALUE = .DST_REC[DQFSL_OVERDRAFT];
1279 1266 2 END;
1280 1267 2
1281 1268 2 ! Issue the ADD function call.
1282 1269 2
1283 1270 2
1284 1271 2 QUOTA_FIB[FIBSW_CNTRLFUNC] = FIBSC_ADD_QUOTA;
1285 1272 2 QUOTA_FIB[FIBSL_CNTRLVAL] = 0;
1286 1273 2 QUOTA_FIB[FIBSL_WCC] = 0;
1287 1274 2 STATUS = $QIOW T[CHAN = .CHANNEL,
1288 1275 2 FUNC = IOS_ACPCONTROL,
1289 1276 2 IOSB = IO_STATUS,
1290 1277 2 P1 = QFIB_DESC,
1291 1278 2 P2 = SRCREC_DESC
1292 1279 2 );
1293 1280 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
1294 1281 2 IF NOT .STATUS
1295 1282 2 THEN ERR_EXIT(DSKQS_ADDERR, .STATUS);
1296 1283 2
1297 1284 2 1
1298 1285 1 END;

```

! end of routine ACT_ADD

			001C 00000	ENTRY	ACT_ADD. Save R2,R3,R4	: 1186
54	00000000G	00	9E 00002	MOVAB	SYSSQIOW, R4	
53	00000000G	00	9E 00009	MOVAB	LIBSTOP, R3	
52	00000000	EF	9E 00010	MOVAB	UIC_FLAGS, R2	
		FE44	C2 B5 00017	TST	.CHANNEL	: 1234
			09 12 0001B	BNEQ	1S	
		00450048	8F DD 0001D	PUSHL	#4522056	: 1235
3FFF	63	86	A2 B1 00023	CALLS	#1 LIBSTOP	
			18:	CMPW	UIC_VALUE+2, #16383	: 1240

04	3B	FFFF	8F	84	08	13	0002C	BEQL	28		1241	
				00450028	A2	B1	0002E	CMPW	UIC_VALUE, #65535			
		63			09	12	00034	BNEQ	33		1242	
		62			8F	DD	00036	28:	PUSHL	#4522024		
		62			01	FB	0003C	CALLS	#1. LIB\$STOP		1248	
		D6	A2		03	E1	0003F	38:	BBC	#3. UIC_FLAGS, 4\$		1249
					04	E0	00043	BBS	#4. UIC_FLAGS, 6\$		1250	
					0C	B0	00047	48:	MOVW	#12. QUOTA_FIB+22		1252
					D8	A2	D4	CLRL	QUOTA_FIB+24		1253	
					D0	A2	D4	CLRL	QUOTA_FIB+16		1254	
					7E	7C	00051	CLRQ	-(SP)		1261	
					48	A2	9F	PUSHAB	DSTREC_DESC			
					7E	D4	00053	CLRL	-(SP)			
					48	A2	9F	PUSHAB	DSTREC_DESC			
					50	A2	9F	PUSHAB	QFIB_DESC			
					FE48	C2	9F	CLRQ	-(SPT)			
		7E	FE44		38	DD	00060	PUSHAB	IO_STATUS			
					C2	3C	00064	PUSHL	#58			
					7E	D4	00068	MOVZWL	CHANNEL, -(SP)			
					OC	FB	0006D	CLRL	-(SP)			
		64			03	E0	00070	CALLS	#12. SYSSQIOW			
		62			A2	A2	00074	BBS	#3. UIC_FLAGS, 5\$		1262	
		8C	A2	AC	04	E0	00079	58:	MOVL	DSF_RECT12. PERM VALUE		1263
		62			A2	D0	0007D	BBS	#4. UIC_FLAGS, 6\$		1264	
		90	A2	B0	08	B0	00082	68:	MOVL	DSF_RECT16. OVER VALUE		1265
		D6	A2		D8	A2	D4	MOVW	#11. QUOTA_FIB+22		1271	
					D0	A2	D4	CLRL	QUOTA_FIB+24		1272	
					7E	7C	00086	CLRL	QUOTA_FIB+16		1273	
					7E	7C	00089	CLRQ	-(SP)		1279	
					40	A2	9F	PUSHAB	SRCREC_DESC			
					50	A2	9F	PUSHAB	QFIB_DESC			
					7E	7C	00096	CLRQ	-(SPT)			
					FE48	C2	9F	PUSHAB	IO_STATUS			
		7E	FE44		38	DD	0009C	PUSHL	#58			
					C2	3C	0009E	MOVZWL	CHANNEL, -(SP)			
					7E	D4	000A3	CLRL	-(SP)			
		64			OC	FB	000A5	CALLS	#12. SYSSQIOW			
		08			50	E9	000A8	BLBC	STATUS, 7\$		1280	
		50		FE48	C2	3C	000AB	MOVZWL	IO_STATUS, STATUS			
		08			50	E8	000B0	BLBS	STATUS, 8\$		1281	
		63		00450078	50	DD	000B3	78:	PUSHL	STATUS		1282
		50			8F	DD	000B5	PUSHL	#4522104			
					02	FB	000BB	CALLS	#2. LIB\$STOP			
					01	DO	000BE	88:	MOVL	#1. R0		
					04	000C1		RET			1285	

; Routine Size: 194 bytes, Routine Base: \$CODES + 0491

```
1300      1286 1 GLOBAL ROUTINE ACT_REMOVE =
1301      1287 1
1302      1288 1 ++
1303      1289 1
1304      1290 1 Functional Description:
1305      1291 1
1306      1292 1 This action routine implements the REMOVE command. It removes the
1307      1293 1 specified entry from the quota file.
1308      1294 1
1309      1295 1 Calling Sequence:
1310      1296 1 standard
1311      1297 1
1312      1298 1 Input Parameters:
1313      1299 1 none
1314      1300 1
1315      1301 1 Implicit Inputs:
1316      1302 1 none
1317      1303 1
1318      1304 1 Output Parameters:
1319      1305 1 none
1320      1306 1
1321      1307 1 Implicit Outputs:
1322      1308 1 none
1323      1309 1
1324      1310 1 Routines Called:
1325      1311 1 none
1326      1312 1
1327      1313 1 Routine Value:
1328      1314 1 none
1329      1315 1
1330      1316 1 Signals:
1331      1317 1 none
1332      1318 1
1333      1319 1 Side Effects:
1334      1320 1 none
1335      1321 1
1336      1322 1 --
1337      1323 1
1338      1324 2 BEGIN
1339      1325 2
1340      1326 2
1341      1327 2 LOCAL STATUS: ! general status value
1342      1328 2
1343      1329 2
1344      1330 2
1345      1331 2
1346      1332 2
1347      1333 2
1348      1334 2
1349      1335 2 IF .CHANNEL EQL 0
1350      1336 2 THEN ERR_EXIT (DSKQS_NODEVICE);
1351      1337 2
1352      1338 2
1353      1339 2
1354      1340 2
1355      1341 2
1356      1342 2
1357      1343 2
1358      1344 2
1359      1345 2
1360      1346 2
1361      1347 2
1362      1348 2
1363      1349 2
1364      1350 2
1365      1351 2
1366      1352 2
1367      1353 2
1368      1354 2
1369      1355 2
1370      1356 2
1371      1357 2
1372      1358 2
1373      1359 2
1374      1360 2
1375      1361 2
1376      1362 2
1377      1363 2
1378      1364 2
1379      1365 2
1380      1366 2
1381      1367 2
1382      1368 2
1383      1369 2
1384      1370 2
1385      1371 2
1386      1372 2
1387      1373 2
1388      1374 2
1389      1375 2
1390      1376 2
1391      1377 2
1392      1378 2
1393      1379 2
1394      1380 2
1395      1381 2
1396      1382 2
1397      1383 2
1398      1384 2
1399      1385 2
1400      1386 2
1401      1387 2
1402      1388 2
1403      1389 2
1404      1390 2
1405      1391 2
1406      1392 2
1407      1393 2
1408      1394 2
1409      1395 2
1410      1396 2
1411      1397 2
1412      1398 2
1413      1399 2
1414      1400 2
1415      1401 2
1416      1402 2
1417      1403 2
1418      1404 2
1419      1405 2
1420      1406 2
1421      1407 2
1422      1408 2
1423      1409 2
1424      1410 2
1425      1411 2
1426      1412 2
1427      1413 2
1428      1414 2
1429      1415 2
1430      1416 2
1431      1417 2
1432      1418 2
1433      1419 2
1434      1420 2
1435      1421 2
1436      1422 2
1437      1423 2
1438      1424 2
1439      1425 2
1440      1426 2
1441      1427 2
1442      1428 2
1443      1429 2
1444      1430 2
1445      1431 2
1446      1432 2
1447      1433 2
1448      1434 2
1449      1435 2
1450      1436 2
1451      1437 2
1452      1438 2
1453      1439 2
1454      1440 2
1455      1441 2
1456      1442 2
1457      1443 2
1458      1444 2
1459      1445 2
1460      1446 2
1461      1447 2
1462      1448 2
1463      1449 2
1464      1450 2
1465      1451 2
1466      1452 2
1467      1453 2
1468      1454 2
1469      1455 2
1470      1456 2
1471      1457 2
1472      1458 2
1473      1459 2
1474      1460 2
1475      1461 2
1476      1462 2
1477      1463 2
1478      1464 2
1479      1465 2
1480      1466 2
1481      1467 2
1482      1468 2
1483      1469 2
1484      1470 2
1485      1471 2
1486      1472 2
1487      1473 2
1488      1474 2
1489      1475 2
1490      1476 2
1491      1477 2
1492      1478 2
1493      1479 2
1494      1480 2
1495      1481 2
1496      1482 2
1497      1483 2
1498      1484 2
1499      1485 2
1500      1486 2
1501      1487 2
1502      1488 2
1503      1489 2
1504      1490 2
1505      1491 2
1506      1492 2
1507      1493 2
1508      1494 2
1509      1495 2
1510      1496 2
1511      1497 2
1512      1498 2
1513      1499 2
1514      1500 2
1515      1501 2
1516      1502 2
1517      1503 2
1518      1504 2
1519      1505 2
1520      1506 2
1521      1507 2
1522      1508 2
1523      1509 2
1524      1510 2
1525      1511 2
1526      1512 2
1527      1513 2
1528      1514 2
1529      1515 2
1530      1516 2
1531      1517 2
1532      1518 2
1533      1519 2
1534      1520 2
1535      1521 2
1536      1522 2
1537      1523 2
1538      1524 2
1539      1525 2
1540      1526 2
1541      1527 2
1542      1528 2
1543      1529 2
1544      1530 2
1545      1531 2
1546      1532 2
1547      1533 2
1548      1534 2
1549      1535 2
1550      1536 2
1551      1537 2
1552      1538 2
1553      1539 2
1554      1540 2
1555      1541 2
1556      1542 2
1557      1543 2
1558      1544 2
1559      1545 2
1560      1546 2
1561      1547 2
1562      1548 2
1563      1549 2
1564      1550 2
1565      1551 2
1566      1552 2
1567      1553 2
1568      1554 2
1569      1555 2
1570      1556 2
1571      1557 2
1572      1558 2
1573      1559 2
1574      1560 2
1575      1561 2
1576      1562 2
1577      1563 2
1578      1564 2
1579      1565 2
1580      1566 2
1581      1567 2
1582      1568 2
1583      1569 2
1584      1570 2
1585      1571 2
1586      1572 2
1587      1573 2
1588      1574 2
1589      1575 2
1590      1576 2
1591      1577 2
1592      1578 2
1593      1579 2
1594      1580 2
1595      1581 2
1596      1582 2
1597      1583 2
1598      1584 2
1599      1585 2
1600      1586 2
1601      1587 2
1602      1588 2
1603      1589 2
1604      1590 2
1605      1591 2
1606      1592 2
1607      1593 2
1608      1594 2
1609      1595 2
1610      1596 2
1611      1597 2
1612      1598 2
1613      1599 2
1614      1600 2
1615      1601 2
1616      1602 2
1617      1603 2
1618      1604 2
1619      1605 2
1620      1606 2
1621      1607 2
1622      1608 2
1623      1609 2
1624      1610 2
1625      1611 2
1626      1612 2
1627      1613 2
1628      1614 2
1629      1615 2
1630      1616 2
1631      1617 2
1632      1618 2
1633      1619 2
1634      1620 2
1635      1621 2
1636      1622 2
1637      1623 2
1638      1624 2
1639      1625 2
1640      1626 2
1641      1627 2
1642      1628 2
1643      1629 2
1644      1630 2
1645      1631 2
1646      1632 2
1647      1633 2
1648      1634 2
1649      1635 2
1650      1636 2
1651      1637 2
1652      1638 2
1653      1639 2
1654      1640 2
1655      1641 2
1656      1642 2
1657      1643 2
1658      1644 2
1659      1645 2
1660      1646 2
1661      1647 2
1662      1648 2
1663      1649 2
1664      1650 2
1665      1651 2
1666      1652 2
1667      1653 2
1668      1654 2
1669      1655 2
1670      1656 2
1671      1657 2
1672      1658 2
1673      1659 2
1674      1660 2
1675      1661 2
1676      1662 2
1677      1663 2
1678      1664 2
1679      1665 2
1680      1666 2
1681      1667 2
1682      1668 2
1683      1669 2
1684      1670 2
1685      1671 2
1686      1672 2
1687      1673 2
1688      1674 2
1689      1675 2
1690      1676 2
1691      1677 2
1692      1678 2
1693      1679 2
1694      1680 2
1695      1681 2
1696      1682 2
1697      1683 2
1698      1684 2
1699      1685 2
1700      1686 2
1701      1687 2
1702      1688 2
1703      1689 2
1704      1690 2
1705      1691 2
1706      1692 2
1707      1693 2
1708      1694 2
1709      1695 2
1710      1696 2
1711      1697 2
1712      1698 2
1713      1699 2
1714      1700 2
1715      1701 2
1716      1702 2
1717      1703 2
1718      1704 2
1719      1705 2
1720      1706 2
1721      1707 2
1722      1708 2
1723      1709 2
1724      1710 2
1725      1711 2
1726      1712 2
1727      1713 2
1728      1714 2
1729      1715 2
1730      1716 2
1731      1717 2
1732      1718 2
1733      1719 2
1734      1720 2
1735      1721 2
1736      1722 2
1737      1723 2
1738      1724 2
1739      1725 2
1740      1726 2
1741      1727 2
1742      1728 2
1743      1729 2
1744      1730 2
1745      1731 2
1746      1732 2
1747      1733 2
1748      1734 2
1749      1735 2
1750      1736 2
1751      1737 2
1752      1738 2
1753      1739 2
1754      1740 2
1755      1741 2
1756      1742 2
1757      1743 2
1758      1744 2
1759      1745 2
1760      1746 2
1761      1747 2
1762      1748 2
1763      1749 2
1764      1750 2
1765      1751 2
1766      1752 2
1767      1753 2
1768      1754 2
1769      1755 2
1770      1756 2
1771      1757 2
1772      1758 2
1773      1759 2
1774      1760 2
1775      1761 2
1776      1762 2
1777      1763 2
1778      1764 2
1779      1765 2
1780      1766 2
1781      1767 2
1782      1768 2
1783      1769 2
1784      1770 2
1785      1771 2
1786      1772 2
1787      1773 2
1788      1774 2
1789      1775 2
1790      1776 2
1791      1777 2
1792      1778 2
1793      1779 2
1794      1780 2
1795      1781 2
1796      1782 2
1797      1783 2
1798      1784 2
1799      1785 2
1800      1786 2
1801      1787 2
1802      1788 2
1803      1789 2
1804      1790 2
1805      1791 2
1806      1792 2
1807      1793 2
1808      1794 2
1809      1795 2
1810      1796 2
1811      1797 2
1812      1798 2
1813      1799 2
1814      1800 2
1815      1801 2
1816      1802 2
1817      1803 2
1818      1804 2
1819      1805 2
1820      1806 2
1821      1807 2
1822      1808 2
1823      1809 2
1824      1810 2
1825      1811 2
1826      1812 2
1827      1813 2
1828      1814 2
1829      1815 2
1830      1816 2
1831      1817 2
1832      1818 2
1833      1819 2
1834      1820 2
1835      1821 2
1836      1822 2
1837      1823 2
1838      1824 2
1839      1825 2
1840      1826 2
1841      1827 2
1842      1828 2
1843      1829 2
1844      1830 2
1845      1831 2
1846      1832 2
1847      1833 2
1848      1834 2
1849      1835 2
1850      1836 2
1851      1837 2
1852      1838 2
1853      1839 2
1854      1840 2
1855      1841 2
1856      1842 2
1857      1843 2
1858      1844 2
1859      1845 2
1860      1846 2
1861      1847 2
1862      1848 2
1863      1849 2
1864      1850 2
1865      1851 2
1866      1852 2
1867      1853 2
1868      1854 2
1869      1855 2
1870      1856 2
1871      1857 2
1872      1858 2
1873
```

```

1357 1343 2 ! Loop for all matching entries in the quota file, making a call to
1358 1344 2 remove each.
1359 1345 2
1360 1346 2
1361 1347 2 QUOTA_FIB[FIBSW_CNTRLFUNC] = FIBSC.REM_QUOTA;
1362 1348 2 QUOTA_FIB[FIBSL_CNTRLVAL] = .UIC_F[AGS];
1363 1349 2 QUOTA_FIB[FIBSL_WCC] = 0;
1364 1350 2
1365 1351 2 INCR J FROM 0
1366 1352 2 DO
1367 1353 2 BEGIN
1368 1354 2
1369 P 1355 2 STATUS = $QIOW (CHAN = .CHANNEL,
1370 P 1356 2 FUNC = IOS_ACPCONTROL,
1371 P 1357 2 IOSB = IO_STATUS,
1372 P 1358 2 P1 = QFIB_DESC,
1373 P 1359 2 P2 = SRCREC_DESC,
1374 P 1360 2 P4 = DSTREC_DESC
1375 1361 2 )
1376 1362 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
1377 1363 2 IF .STATUS
1378 1364 2 THEN
1379 1365 4 BEGIN
1380 1366 4 IF .STATUS EQ SSS_DVRDSKQUOTA
1381 1367 4 THEN ERR_MESSAGE (DSKQS_INUSE,
1382 1368 4 .(DST_REC[DQFSL_UIC])<16,16>,
1383 1369 4 .(DST_REC[DQFSL_UIC])<00,16>,
1384 1370 4 .DST_REC[DQFSL_USAGE]);
1385 1371 4 END
1386 1372 3 ELSE
1387 1373 4 BEGIN
1388 1374 4 IF .STATUS EQ SSS_NODISKQUOTA
1389 1375 4 AND .J NEQ 0
1390 1376 4 THEN EXITLOOP;
1391 1377 4 ERR_EXIT (DSKQS_REMOVEERR, .STATUS);
1392 1378 4 END;
1393 1379 4
1394 1380 4 IF NOT .UIC_FLAGS[WILD_GROUP]
1395 1381 4 AND NOT .UIC_FLAGS[WILD_MEMBER]
1396 1382 4 THEN EXITLOOP; ! done if no wild cards
1397 1383 4
1398 1384 2 END; ! end of loop
1399 1385 2
1400 1386 1 ! END: ! end of routine ACT_REMOVE
1401 1387 1

```

55 00000000G	00 003C	00000	.ENTRY ACT REMOVE, Save R2,R3,R4,R5	: 1286
56 00000000,	EF 9E	00002	MOVAB LIB\$STOP, R5	
	FE44 C4	00009	MOVAB UIC_FLAGS, R6	
	09 B5	00010	TSTW CHANNEL	: 1334
65 00450048	09 12	00014	BNEQ 1S	
	8F DD	00016	PUSHL #4522056	: 1335
	01 FB	0001C	CALLS #1, LIB\$STOP	

3FFF	8F	86	A4	B1	0001F	18:	CMPW	UIC_VALUE+2, #16383	1340	
FFFF	64	84	A4	B1	0002A	28:	BNEQ	28		
	8F		02	88	00027		BISB2	#2, UIC FLAGS	1341	
			03	12	00030	28:	CMPW	UIC_VALUE, #65535		
D6	64		01	88	00032		BNEQ	38		
D6	A4		0E	B0	00035	38:	BISB2	#1, UIC FLAGS	1347	
D8	A4		64	D0	00039		MOVW	#14, QUOTA_FIB+22	1348	
		DD	A4	D4	0003D		MOVL	UIC_FLAGS, -QUOTA_FIB+24	1349	
			53	D4	00040		CLRL	QUOTA_FIB+16	1351	
			7E	7C	00042	48:	CLRL	J	1361	
			48	A4	9F	00044	CLRQ	-(SP)		
				7E	D4	00047	PUSHAB	DSTREC_DESC		
				40	A4	9F	00049	CLRL	-(SP)	
				50	A4	9F	0004C	PUSHAB	SRCREC_DESC	
					7E	7C	0004F	PUSHAB	QFIB_DESC	
							CLRQ	-(SP)		
			FE48	C4	9F	00051	PUSHAB	IO_STATUS		
				38	DD	00055	PUSHL	#58		
			7E	FE44	C4	3C	00057	MOVZWL	CHANNEL, -(SP)	
					7E	D4	0005C	CLRL	-(SP)	
00000000G	00			0C	FB	0005E	CALLS	#12, SYSSQIOW		
	52			50	D0	00065	MOVL	RO_STATUS		
	28			52	F9	00068	BLBC	STATUS, 58	1362	
	52		FE48	C4	3C	0006B	MOVZWL	IO_STATUS, STATUS		
	23			52	E9	00070	BLBC	STATUS, 58	1363	
00000669	8F			52	D1	00073	CMPL	STATUS, #1641	1366	
				32	12	0007A	BNEQ	78		
				A8	A4	DD	PUSHL	DST_REC+8	1370	
			7E	A4	A4	3C	MOVZWL	DST_REC+4, -(SP)		
			7E	A6	A4	3C	MOVZWL	DST_REC+6, -(SP)		
		00450098		8F	DD	00087	PUSHL	#4522136		
00000000G	00			04	FB	0008D	CALLS	#4, LIB\$SIGNAL		
				18	11	00094	BRB	78	1363	
000003E4	BF			52	D1	00096	58:	CMPL	STATUS, #996	1374
				04	12	0009D	BNEQ	68		
				53	D5	0009F	TSTL	J	1375	
				1A	12	000A1	BNEQ	98		
				52	DD	000A3	68:	PUSHL	STATUS	1377
				8F	DD	000A5	PUSHL	#4522112		
03	65			02	FB	000AB	CALLS	#2, LIB\$STOP		
	64			01	E0	000AE	78:	BBS	#1, UIC FLAGS, 88	1380
	08			64	E9	000B2	BLBC	UIC_FLAGS, 98	1381	
85	53	7FFFFFFF		8F	F3	000B5	88:	AOBLEQ	#2177483647, J, 48	1351
	50			01	D0	000BD	98:	MOVL	#1, RO	1387
				04	000C0		RET			

; Routine Size: 193 bytes, Routine Base: SCODES + 0553

1403 1388 1 GLOBAL ROUTINE ACT_SHOW =
1404 1389 1
1405 1390 1
1406 1391 1
1407 1392 1
1408 1393 1
1409 1394 1
1410 1395 1
1411 1396 1
1412 1397 1
1413 1398 1
1414 1399 1
1415 1400 1
1416 1401 1
1417 1402 1
1418 1403 1
1419 1404 1
1420 1405 1
1421 1406 1
1422 1407 1
1423 1408 1
1424 1409 1
1425 1410 1
1426 1411 1
1427 1412 1
1428 1413 1
1429 1414 1
1430 1415 1
1431 1416 1
1432 1417 1
1433 1418 1
1434 1419 1
1435 1420 1
1436 1421 1
1437 1422 1
1438 1423 1
1439 1424 1
1440 1425 1
1441 1426 2 BEGIN
1442 1427 2
1443 1428 2 BIND
1444 1429 2 LISTING_HEADER = DESCRIPTOR ('UIC Usage Permanent Quota Overdraft Limit'),
1445 1430 2 MULTI_FORMAT_1 = DESCRIPTOR ('!18<!AS!>!13<!UL!>!18<!UL!>!13<!UL!>'),
1446 1431 2 MULTI_FORMAT_2 = DESCRIPTOR ('!AS!/!18+!13<!UL!>!18<!UL!>!13<!UL!>'),
1447 1432 2 SINGLE_FORMAT = DESCRIPTOR ('UIC !AS has !UL blocks used!/of !UL authorized, !UL permitted overdra
1448 1433 2
1449 1434 2
1450 1435 2 LOCAL
1451 1436 2 UIC_DESC : SBBLOCK [DSCBC S_BLN]. ! Descr for alpha UIC
1452 1437 2 FORMATTED_UIC : VECTOR [67,BYTE]. ! Alpha UIC storage
1453 1438 2 STATUS; ! general status value
1454 1439 2
1455 1440 2 EXTERNAL ROUTINE
1456 1441 2 LIBSPUT_OUTPUT : ADDRESSING_MODE (GENERAL);
1457 1442 2
1458 1443 2
1459 1444 2 ! Verify that a channel is open.

```
1460 1445 2 !
1461 1446 2
1462 1447 2 IF .CHANNEL EQ 0
1463 1448 2 THEN ERR_EXIT (DSKQS_NODEVICE);
1464 1449 2
1465 1450 2 ! Set any appropriate wildcard indicators.
1466 1451 2
1467 1452 2
1468 1453 2 IF .UIC_VALUE<16,16> EQ UIC_SK_WILD_GROUP THEN UIC_FLAGS[WILD_GROUP] = 1;
1469 1454 2 IF .UIC_VALUE<0,16> EQ UIC_SK_WILD_MEMBER THEN UIC_FLAGS[WILD_MEMBER] = 1;
1470 1455 2
1471 1456 2 ! Loop for all matching entries in the quota file, making a call to
1472 1457 2 ! examine each.
1473 1458 2
1474 1459 2
1475 1460 2 QUOTA_FIB[FIBSW_CNTRLFUNC] = FIBSC_EXA_QUOTA;
1476 1461 2 QUOTA_FIB[FIBSL_CNTRLVAL] = .UIC_FLAGS;
1477 1462 2 QUOTA_FIB[FIBSL_WCC] = 0;
1478 1463 2
1479 1464 2 IF .UIC_FLAGS[WILD_GROUP]
1480 1465 2 OR .UIC_FLAGS[WILD_MEMBER]
1481 1466 2 THEN LIBSPUT_OUTPUT (LISTING_HEADER);
1482 1467 2
1483 1468 2 INCR J FROM 0
1484 1469 2 DO
1485 1470 2 BEGIN
1486 1471 2
P 1472 3 STATUS = $QIOW (CHAN = .CHANNEL,
1473 3 FUNC = IOS_ACPCONTROL,
1474 3 IOSB = IO_STATUS,
1475 3 P1 = QFIB_DESC,
1476 3 P2 = SRCREC_DESC,
1477 3 P4 = DSTREC_DESC
1478 3 )
1479 3
1480 3 IF .STATUS THEN STATUS = .IO_STATUS[0];
1481 3 IF NOT .STATUS
1482 4 THEN
1483 4 BEGIN
1484 4 IF .STATUS EQ SSS_NODISKQUOTA
1485 4 AND .J NEQ 0
1486 4 THEN EXITLOOP;
1487 4 ERR_EXIT (DSKQS_EXAMINERR, .STATUS);
1488 3
1489 3 ! Format a listing line and output it.
1490 3
1491 3
1492 1492 3 UIC_DESC[DSC$W_LENGTH] = 67;
1493 1493 3 UIC_DESC[DSC$A_POINTER] = FORMATTED_UIC;
1494 1494 3 SFAO ($DESCRIPTOR ('!%'), UIC_DESC, UIC_DESC,
1495 1495 3 .DST_REC[DQFSE_UIC]);
1496 1496 3
1497 1497 3 OUTPUT_DESC[0] = OUTPUT_LENGTH;
1498 1498 3 SFAO (
1499 1499 3
1500 1500 3
P 1501 1501 3
1502 1502 3
1503 1503 3
1504 1504 3
1505 1505 3
1506 1506 3
1507 1507 3
1508 1508 3
1509 1509 3
P 1510 1510 3
1511 1511 3
1512 1512 3
1513 1513 3
1514 1514 3
1515 1515 3
P 1516 1516 3
P 1501 1501 3
```

```

1517 P 1502 OR .UIC FLAGS[WILD MEMBER]
1518 P 1503 THEN (IF .UIC DESC[DSC$W_LENGTH] LSS 18
1519 P 1504 THEN MULTI FORMAT
1520 P 1505 ELSE MULTI FORMAT_2)
1521 P 1506 ELSE SINGLE FORMAT),
1522 P 1507 OUTPUT_DESC[0];
1523 P 1508 OUTPUT_DESC[0];
1524 P 1509 UIC_DESC
1525 P 1510 .DST_REC[DQFSL_USAGE],
1526 P 1511 .DST_REC[DQFSL_PERMQUOTA],
1527 P 1512 .DST_REC[DQFSL_OVERDRAFT]
1528 P 1513 );
1529 P 1514 LIBSPUT_OUTPUT (OUTPUT_DESC[0]);
1530 P 1515 IF NOT .UIC FLAGS[WILD GROUP]
1531 P 1516 AND NOT .UIC FLAGS[WILD_MEMBER]
1532 P 1517 THEN EXITLOOP; ! done if no wild cards
1533 P 1518
1534 P 1519 ! end of loop
1535 P 1520
1536 P 1521 END;
1537 P 1522 ! end of routine ACT_SHOW
1538 P 1523
1539 P 1524 1 END;

```

.PSECT SPLIT\$,NOWRT,NOEXE,2

20 20 20 20 20 20 20 43 49 55 20 20 20 20 20 0031C P.AAK:	.ASCII \ UIC Usage Permanent\
20 20 20 20 20 20 74 65 67 61 73 55 20 20 20 0032B	
72 64 72 65 76 4F 20 20 20 61 74 6F 75 51 20 0033A	
74 69 60 69 4C 20 74 66 61 00000040, 00000000, 00344	.ASCII \ Quota Overdraft Limit\
55 31 3C 33 31 21 3E 55 21 3C 41 21 3C 38 31 21 00353	P.AAJ: .LONG 64
33 31 21 3E 21 4C 55 21 3C 38 31 21 3E 21 4C 00360	.ADDRESS P.AAK
55 21 3C 33 31 21 4C 55 21 3C 38 31 21 3E 21 4C 00364	P.AAM: .ASCII \!AS!>!13<!UL!>!18<!UL!>!13<!UL!>\
21 3C 33 31 21 4C 55 21 3C 38 31 21 2F 21 53 41 21 00373	P.AAL: .LONG 36
31 21 3C 33 31 21 4C 55 21 3C 38 31 21 3E 21 4C 00382	.ADDRESS P.AAM
21 3C 33 31 21 4C 55 21 3C 38 31 21 2F 21 53 41 21 00388	P.AAO: .ASCII \!AS!/!18+ !13<!UL!>!18<!UL!>!13<!UL!>\
31 21 3C 33 31 21 4C 55 21 3C 38 31 21 4C 55 21 3C 33 0039F	
	P.AAE: .BLKB 3
	P.AAN: .LONG 37
	.ADDRESS P.AAO
4C 55 21 20 73 61 68 20 53 41 21 20 43 49 55 003BC	P.AAQ: .ASCII \UIC !AS has !UL blocks used!/of !UL auth\
6F 2F 21 64 65 73 75 20 73 68 63 6F 6C 62 20 003C0	
68 74 75 61 20 4C 55 21 20 66 003CF	
72 65 70 20 4C 55 21 20 2C 64 65 7A 69 72 6F 003DE	
66 61 72 64 72 65 76 6F 20 64 65 74 74 69 6D 003E8	.ASCII \orized, !UL permitted overdraft.\
	003F7
	2E 74 00406
	00000048, 00000000, 00408 P.AAP: .LONG 72
	49 25 21 0040C .ADDRESS P.AAQ
	00410 P.AAS: .ASCII \!Z\
	00413 .BLKB 1
	00000003 00414 P.AAR: .LONG 3

00000000* 00418 .ADDRESS P.AAS
 LISTING HEADER= P.AAJ
 MULTI_FORMAT_1= P.AAL
 MULTI_FORMAT_2= P.AAN
 SINGLE_FORMAT= P.AAP
 .EXTRN LIB\$PUT_OUTPUT. SYSSFAO
 .PSECT SCODE\$, NOWRT, 2
 01FC 00000 .ENTRY ACT SHOW, Save R2,R3,R4,R5,R6,R7,R8
 00 9E 00002 MOVAB SYSSFAO, R8
 00 9E 00009 MOVAB LIB\$PUT_OUTPUT, R7
 00 9E 00010 MOVAB LIB\$STOP, R6
 EF 9E 00017 MOVAB LISTING HEADER, R5
 EF 9E 0001E MOVAB UIC FLAGS, R4
 AE 9E 00025 MOVAB -767SP), SP
 C4 85 00029 TSTW CHANNEL
 09 12 0002D BNEQ 18
 8F DD 0002F PUSHL #4522056
 01 FB 00035 CALLS #1, LIB\$STOP
 A4 B1 00038 1\$: CMPW UIC_VALUE+2, #16383
 03 12 0003E BNEQ 28
 02 88 00040 BISB2 #2, UIC FLAGS
 A4 B1 00043 2\$: CMPW UIC_VALUE, #65535
 03 12 00049 BNEQ 38
 01 88 00048 BISB2 #1, UIC FLAGS
 0C B0 0004E 3\$: MOVW #12, QUOTA_FIB+22
 64 D0 00052 MOVL UIC FLAGS, QUOTA_FIB+24
 A4 D4 00056 CLRL QUOTA_FIB+16
 01 E0 00059 BBS #1, UIC FLAGS, 48
 64 E9 0005D BLBC UIC_FLAGS, 58
 55 DD 00060 4\$: PUSHL R5
 01 FB 00062 CALLS #1, LIB\$PUT_OUTPUT
 53 D4 00065 5\$: CLPL J
 7E 7C 00067 6\$: CLRQ -(SP)
 A4 9F 00069 PUSHAB DSTREC_DESC
 7E D4 0006C CLRL -(SP)
 A4 9F 0006E PUSHAB SRCREC_DESC
 A4 9F 00071 PUSHAB QFIB_DESC
 7E 7C 00074 CLRQ -(SP)
 C4 9F 00076 PUSHAB IO STATUS
 38 DD 0007A PUSHL #58
 C4 3C 0007C MOVZWL CHANNEL, -(SP)
 7E D4 00081 CLRL -(SP)
 0C FB 00083 CALLS #12, SYSSQIOW
 50 D0 0008A MOVL R0, STATUS
 52 E9 0008D BLBC STATUS, 78
 C4 3C 00090 MOVZWL IO STATUS, STATUS
 52 E8 00095 BLBS STATUS, 98
 52 D1 00098 7\$: CMPL STATUS, #996
 04 12 0009F BNEQ 88
 53 D5 000A1 TSTL J
 77 12 000A3 BNEQ 158
 52 DD 000A5 8\$: PUSHL STATUS
 8F DD 000A7 PUSHL #4522128
 02 FB 000AD CALLS #2, LIB\$STOP

		44	AE	43	8F	9B	000B0	98:	MOVZBW	#67, UIC_DESC	1492
		48	AE		6E	9E	000B5		MOVAB	FORMATTED_UIC, UIC_DESC+4	1493
				A4	A4	DD	000B9		PUSHL	DST_REC+4	1496
				48	AE	9F	000BC		PUSHAB	UIC_DESC	
				4C	AE	9F	000BF		PUSHAB	UIC_DESC	
				00B8	C5	9F	000C2		PUSHAB	P_AAR	001
		FF60	68		04	FB	000C6		CALLS	#4, SYSSFAO	
			C4	B4	8F	9A	000C9		MOVZBL	#132, OUTPUT_DESC	
			7E	AC	A4	7D	000CF		MOVO	DST_REC+12, =(SP)	1498
				A8	A4	DD	000D3		PUSHL	DST_REC+8	1513
				50	AE	9F	000D6		PUSHAB	UIC_DESC	
				FF60	C4	9F	000D9		PUSHAB	OUTPUT_DESC	
				FF60	C4	9F	000DD		PUSHAB	OUTPUT_DESC	
03			64		01	E0	000E1		BBS	#1, UIC_FLAGS, 10\$	
			12		64	E9	000E5		BLBC	UIC_FLAGS, 12\$	
			12	5C	AE	B1	000E8	10\$:	CMPW	UIC_DESC, #18	
					06	1E	000EC		BGEQU	11\$	
			50	2C	A5	9E	000EE		MOVAB	MULTI_FORMAT_1, R0	
					0B	11	000F2		BRB	13\$	
			50	5C	A5	9E	000F4	11\$:	MOVAB	MULTI_FORMAT_2, R0	
				00AC	C5	9E	000FA	12\$:	BRB	13\$	
					50	DD	000FF	13\$:	MOVAB	SINGLE_FORMAT, R0	
			68		07	FB	00101		PUSHL	R0	
				FF60	C4	9F	00104		CALLS	#7, SYSSFAO	
			67		01	FB	00108		PUSHAB	OUTPUT_DESC	1515
	03		64		01	E0	0010B		CALLS	#1, LIB\$PUT_OUTPUT	
			0A		64	E9	0010F		BBS	#1, UIC_FLAGS, 14\$	1517
FF4B	53		01	7FFFFFFF	8F	F1	00112	14\$:	BLBC	UIC_FLAGS, 15\$	1518
			50		01	D0	0011C	15\$:	ACBL	#2147483647, #1, J, 6\$	1468
					04	0011F		MOVL	#1, R0	1524	
								RET			

: Routine Size: 288 bytes, Routine Base: \$CODE\$ + 0614

```
1541      1525 1 GLOBAL ROUTINE ACT_MODIFY =  
1542      1526 1  
1543      1527 1 :++  
1544      1528 1  
1545      1529 1 Functional Description:  
1546      1530 1  
1547      1531 1 This action routine implements the MODIFY command. It modifies the  
1548      1532 1 specified entry of the quota file as specified.  
1549      1533 1  
1550      1534 1 Calling Sequence:  
1551      1535 1 standard  
1552      1536 1  
1553      1537 1 Input Parameters:  
1554      1538 1 none  
1555      1539 1  
1556      1540 1 Implicit Inputs:  
1557      1541 1 none  
1558      1542 1  
1559      1543 1 Output Parameters:  
1560      1544 1 none  
1561      1545 1  
1562      1546 1 Implicit Outputs:  
1563      1547 1 none  
1564      1548 1  
1565      1549 1 Routines Called:  
1566      1550 1 none  
1567      1551 1  
1568      1552 1 Routine Value:  
1569      1553 1 none  
1570      1554 1  
1571      1555 1 Signals:  
1572      1556 1 none  
1573      1557 1  
1574      1558 1 Side Effects:  
1575      1559 1 none  
1576      1560 1  
1577      1561 1 --  
1578      1562 1  
1579      1563 2 BEGIN  
1580      1564 2  
1581      1565 2  
1582      1566 2 LOCAL STATUS;           ! general status value  
1583      1567 2  
1584      1568 2  
1585      1569 2  
1586      1570 2 : Verify that a channel is open.  
1587      1571 2  
1588      1572 2  
1589      1573 2 IF .CHANNEL EQL 0  
1590      1574 2 THEN ERR_EXIT (DSKOS_NODEVICE);  
1591      1575 2  
1592      1576 2 : Set any appropriate wildcard indicators.  
1593      1577 2  
1594      1578 2  
1595      1579 2 IF .UIC_VALUE<16,16> EQL UIC$K_WILD_GROUP THEN UIC_FLAGS[WILD_GROUP] = 1;  
1596      1580 2 IF .UIC_VALUE<0,16> EQL UIC$K_WILD_MEMBER THEN UIC_FLAGS[WILD_MEMBER] = 1;  
1597      1581 2
```

```

1598 1582 2 1 Loop for all matching entries in the quota file, making a call to
1599 1583 2 1 modify each.
1600 1584 2 1
1601 1585 2 1
1602 1586 2 1 QUOTA_FIB[FIBSW_CNTRLFUNC] = FIBSC_MCD_QUOTA;
1603 1587 2 1 QUOTA_FIB[FIBSL_CNTRLVAL] = .UIC_F[AGS];
1604 1588 2 1 QUOTA_FIB[FIBSL_WCC] = 0;
1605 1589 2 1
1606 1590 2 1 INCR J FROM 0
1607 1591 2 1 DO
1608 1592 2 1 BEGIN
1609 1593 2 1
P 1610 1594 2 1 STATUS = SQIOW (CHAN = .CHANNEL,
1611 P 1595 2 1 FUNC = IOS_ACPCONTROL,
1612 P 1596 2 1 IOSB = IO_STATUS,
1613 P 1597 2 1 P1 = QFTB_DESC,
1614 P 1598 2 1 P2 = SRCREC_DESC,
1615 P 1599 2 1 P4 = DSTREC_DESC
1616 1600 2 1 );
1617 1601 2 1 IF .STATUS THEN STATUS = .IO_STATUS[0];
1618 1602 2 1 IF .STATUS
1619 1603 2 1 THEN
1620 1604 2 1 BEGIN
1621 1605 2 1 IF .STATUS EQ SSS_OVRDSKQUOTA
1622 P 1606 2 1 THEN ERR_MESSAGE (DSKQS_INUSE,
1623 P 1607 2 1 .(DST_REC[DQFSL_UIC])<16,16>,
1624 P 1608 2 1 .(DST_REC[DQFSL_UIC])<00,16>,
1625 P 1609 2 1 .DST_REC[DQFSL_USAGE]);
1626 1610 2 1 END
1627 1611 2 1 ELSE
1628 1612 2 1 BEGIN
1629 1613 2 1 IF .STATUS EQ SSS_NODISKQUOTA
1630 1614 2 1 AND .J NEQ 0
1631 1615 2 1 THEN EXITLOOP;
1632 1616 2 1 ERR_EXIT (DSKQS_MODIFYERR, .STATUS);
1633 1617 2 1 END;
1634 1618 2 1
1635 1619 2 1 IF NOT .UIC_FLAGS[WILD_GROUP]
1636 1620 2 1 AND NOT .UIC_FLAGS[WILD_MEMBER]
1637 1621 2 1 THEN EXITLOOP; ! done if no wild cards
1638 1622 2 1
1639 1623 2 1 END; ! end of loop
1640 1624 2 1
1641 1625 2 1
1642 1626 1 1 END; ! end of routine ACT_MODIFY

```

55 0000000G	00 9E 00002	.ENTRY ACT MODIFY Save R2,R3,R4,R5	: 1525
54 00000C00	EF 9E 00009	MOVAB LIB\$STOP, R5	
	FE44 C4 B5 00010	MOVAB UIC_FLAGS, R4	: 1573
	09 12 00014	TSTW CHANNEL	
65 00450048	8F DD 00016	BNEQ 1S	: 1574
	01 FB 0001C	PUSHL #4522056	
		CALLS #1, LIB\$STOP	

3FFF	8F	86	A4	B1	0001F	18:	CMPW	UIC_VALUE+2, #16383	: 1579
	64		03	12	00025		BNEQ	28	
FFFF	8F	84	A4	B1	0002A	28:	BISB2	#2 UIC FLAGS	: 1580
	03		03	12	00030		CMPW	UIC_VALUE, #65535	
	64		01	88	00032		BNEQ	38	
D6	A4		00	80	00035	38:	BISB2	#1 UIC FLAGS	: 1586
D8	A4	D0	64	D0	00039		MOVW	#13, QUOTA_FIB+22	: 1587
			A4	D4	0003D		MOVL	UIC FLAGS, QUOTA_FIB+24	: 1588
			53	D4	00040		CLRL	QUOTA_FIB+16	: 1590
			7E	7C	00042	48:	CLRL	J	: 1600
			48	A4	9F	00044	CLRQ	-(SP)	
				7E	D4	00047	PUSHAB	DSTREC_DESC	
				40	A4	9F	00049	CLRL	-(SP)
				50	A4	9F	0004C	PUSHAB	SRCREC_DESC
					7E	7C	0004F	PUSHAB	OFIB_DESC
					FE48	C4	9F	00051	CLRQ
						38	DD	00055	-(SP)
						C4	3C	00057	PUSHAB
						7E	D4	0005C	IO_STATUS
							7E	FB	#56
00000000G	00		0C		0005E		MOVZWL	CHANNEL, -(SP)	
	52		50	D0	00065		CLRL	-(SP)	
	2B		52	E9	00068		CALLS	#12, SYSSQIOW	: 1601
	52	FE48	C4	3C	0006B		MOVL	RO, STATUS	
	23		52	E9	00070		BLBC	STATUS, 58	
00000669	8F		52	D1	00073		MOVZWL	IO_STATUS, STATUS	: 1602
			32	12	0007A		BLBC	STATUS, 58	
			A8	A4	0007C		CMPL	STATUS, #1641	: 1605
	7E		A4	3C	0007F		BNEQ	78	
	7E	A6	A4	3C	00083		PUSHL	DST_REC+8	: 1609
00000000G	00	00450098	8F	DD	00087		MOVZWL	DST_REC+4, -(SP)	
			04	FB	0008D		MOVZWL	DST_REC+6, -(SP)	
			18	11	00094		PUSHL	#4522136	
000003E4	8F		52	D1	00096	58:	CALLS	#4, LIB\$SIGNAL	: 1602
			04	12	0009D		BRB	78	
			53	D5	0009F		CMPL	STATUS, #996	: 1613
			1A	12	000A1		BNEQ	68	
			52	DD	000A3	68:	TSTL	J	: 1614
			8F	DD	000A5		BNEQ	98	
03	65	00450088	02	FB	000AB		PUSHL	STATUS	: 1616
	64		01	E0	000AE	78:	CALLS	#4522120	
	08		64	E9	000B2		BBS	#2, LIB\$STOP	: 1619
85	53	7FFFFFFF	8F	F3	000B5	88:	BLBC	#1, UIC FLAGS, 88	: 1620
	50		01	DD	000BD	98:	AOBLEQ	UIC FLAGS, 98	: 1590
			04	000C0			MOVL	#217483647, J, 48	
							RET	#1, RO	: 1626

; Routine Size: 193 bytes, Routine Base: SCODES + 0734

```
1644 1627 1 GLOBAL ROUTINE ACT_REBUILD =  
1645 1628 1  
1646 1629 1 !++  
1647 1630 1  
1648 1631 1 Functional Description:  
1649 1632 1  
1650 1633 1 This routine implements the REBUILD command. It scans the index file  
1651 1634 1 of each volume in the volume set and constructs a table of UIC's  
1652 1635 1 and blocks used. It then updates the usage data in the quota file,  
1653 1636 1 creating entries as needed so that all UIC's using blocks are listed.  
1654 1637 1  
1655 1638 1 Calling Sequence:  
1656 1639 1 standard  
1657 1640 1  
1658 1641 1 Input Parameters:  
1659 1642 1 none  
1660 1643 1  
1661 1644 1 Implicit Inputs:  
1662 1645 1 none  
1663 1646 1  
1664 1647 1 Output Parameters:  
1665 1648 1 none  
1666 1649 1  
1667 1650 1 Implicit Outputs:  
1668 1651 1 none  
1669 1652 1  
1670 1653 1 Routines Called:  
1671 1654 1 none  
1672 1655 1  
1673 1656 1 Routine Value:  
1674 1657 1 none  
1675 1658 1  
1676 1659 1 Signals:  
1677 1660 1 none  
1678 1661 1  
1679 1662 1 Side Effects:  
1680 1663 1 none  
1681 1664 1  
1682 1665 1 !--  
1683 1666 1  
1684 1667 2 BEGIN  
1685 1668 2  
1686 1669 2 LOCAL STATUS; : general status value  
1687 1670 2  
1^88 1671 2  
1689 1672 2 EXTERNAL ROUTINE REBUILD : ADDRESSING_MODE (GENERAL); ! routine to do actual rebuild  
1690 1673 2  
1691 1674 2  
1692 1675 2  
1693 1676 2 ! Verify that a channel is open.  
1694 1677 2  
1695 1678 2  
1696 1679 2 IF .CHANNEL EQL 0  
1697 1680 2 THEN ERR_EXIT (DSKQS_NODEVICE);  
1698 1681 2  
1699 1682 2 ! Enable the quota file, just in case it is off.  
1700 1683 2
```

```

1701 1684 2
1702 1685 2 QUOTA_FIB[FIBSW_DID_NUM] = FIDSC_MFD;
1703 1686 2 QUOTA_FIB[FIBSW_DID_SEQ] = FIDSC_MFD;
1704 1687 2 QUOTA_FIB[FIBSW_DID_RVN] = 1;
1705 1688 2 QUOTA_FIB[FIBSW_CNTRLFUNC] = FIBSC_ENA_QUOTA;
1706 P 1689 2 STATUS = $QIOW TCHAN = .CHANNEL,
1707 P 1690 2 FUNC = IOS_ACPCONTROL,
1708 P 1691 2 IOSB = IO_STATUS,
1709 P 1692 2 P1 = QFIB_DESC,
1710 P 1693 2 P2 = QFILE_NAME
1711 1694 2
1712 1695 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
1713 1696 2 IF NOT .STATUS
1714 1697 2 AND .STATUS NEQ SSS_QACTIVE
1715 1698 2 THEN ERR_EXIT (DSKQS_ACTERR, .STATUS);
1716 1699 2
1717 1700 2 ! Now call the rebuild routine.
1718 1701 2
1719 1702 2
1720 1703 2 REBUILD (.CHANNEL, 1);
1721 1704 2
1722 1705 2 1
1723 1706 1 END;

```

: end of routine ACT_REBUILD

.EXTRN REBUILD

				.ENTRY ACT REBUILD, Save R2,R3	1627
				MOVAB LIB\$STOP, R3	
				MOVAB CHANNEL, R2	1679
				TSTW CHANNEL	
				BNEQ 1\$	
				PUSHL #4522056	1680
				CALLS #1 LIB\$STOP	
				MOVL #262148, QUOTA_FIB+10	1685
				MOVW #1, QUOTA_FIB+T4	1687
				MOVW #9, QUOTA_FIB+22	1688
				CLRQ -(SP)	1694
				CLRQ -(SP)	
				PUSHAB QFILE_NAME	
				PUSHAB QFIB_DESC	
				CLRQ -(SP)	
				PUSHAB IO_STATUS	
				PUSHL #58	
				MOVZWL CHANNEL, -(SP)	
				CRL -(SP)	
				CALLS #12, SYSSQIOW	1695
				BLBC STATUS, 28	
				MOVZWL IO_STATUS STATUS	1696
				BLBS STATUS, 38	1697
				CMPL STATUS, #972	1698
				BEQL 38	
				PUSHL STATUS	
				PUSHL #4522088	
				CALLS #2, LIB\$STOP	
				PUSHL #1	1703

DISKQUOTA
V04-000

5 14
15-Sep-1984 23:38:38 14-Sep-1984 12:19:46 VAX-11 Bliss-32 v4.0-742
[DISKQ.SRC]DISKQUOTA.B32:1

Page 68
(18)

00000000G 7E
00
50

62 3C 0006F
02 FB 00072
01 D0 00079
04 0007C

MOVZWL CHANNEL, -(SP)
CALLS #2, REBUILD
MOVL #1, R0
RET

1706
⋮
⋮

; Routine Size: 125 bytes, Routine Base: \$CODE\$ + 07F5

1725 1707 1 GLOBAL ROUTINE ACT_HELP =
1726 1708 1 ++
1727 1709 1
1728 1710 1 Functional Description:
1729 1711 1
1730 1712 1 This routine is the DISKQUOTA help facility, and will display
1731 1713 1 useful and informative explanations of the DISKQUOTA facility.
1732 1714 1
1733 1715 1 To speed things up, the help library is opened only once, and
1734 1716 1 is closed by the OS during image rundown.
1735 1717 1
1736 1718 1 Calling Sequence:
1737 1719 1 standard
1738 1720 1
1739 1721 1 Input Parameters:
1740 1722 1 none
1741 1723 1
1742 1724 1 Implicit Inputs:
1743 1725 1
1744 1726 1 This routine expects the keys used to access the help text to
1745 1727 1 be in KEY_VECTOR[0..MAX_KEYS].
1746 1728 1
1747 1729 1 Output Parameters:
1748 1730 1 none
1749 1731 1
1750 1732 1 Implicit Outputs:
1751 1733 1
1752 1734 1 The help text will be printed on SYSSOUTPUT.
1753 1735 1
1754 1736 1 Routines Called:
1755 1737 1
1756 1738 1 LBR\$INI_CONTROL
1757 1739 1 LBR\$OPEN
1758 1740 1 LBR\$GET_HELP
1759 1741 1
1760 1742 1 Routine Value:
1761 1743 1 none
1762 1744 1
1763 1745 1 Signals:
1764 1746 1 none
1765 1747 1
1766 1748 1 Side Effects:
1767 1749 1 none
1768 1750 1
1769 1751 1 --
1770 1752 1
1771 1753 2 BEGIN
1772 1754 2
1773 1755 2 EXTERNAL ROUTINE
1774 1756 2
1775 1757 2 LBR\$INI_CONTROL : ADDRESSING_MODE(GENERAL),
1776 1758 2 LBR\$OPEN : ADDRESSING_MODE(GENERAL),
1777 1759 2 LBR\$GET_HELP : ADDRESSING_MODE(GENERAL);
1778 1760 2
1779 1761 2 BIND
1780 1762 2
1781 1763 2 HELP_DEFNAME = DESCRIPTOR ('SYSSHELP:.HLB'), ! default helpfile name

EX
--
DI
DI
CL
LO
SY
LI
SY
--
CL
--
CL

```

: 1782      1764 2 LIBRARY_NAME = DESCRIPTOR ('DISKQUOTA');    ! HELP text library
: 1783      1765 2 OWN
: 1784      1766 2
: 1785      1767 2
: 1786      1768 2 HELP_FUNCTION : INITIAL (LBRSC_READ),
: 1787      1769 2 HELP_TYPE   : INITIAL (LBRSC_TYP_HLP).      ! declare lib a HELP lib
: 1788      1770 2 HELP_LIBINDEX : LONG.                      pointer to lib index
: 1789      1771 2 LIBRARY_OPEN  : LONG;                      used as a boolean
: 1790
: 1791      1772 2 LOCAL
: 1792      1773 2
: 1793      1774 2 STATUS:                                     ! used as boolean
: 1794      1775 2
: 1795      1776 2
: 1796      1777 2
: 1797      1778 2 | Check to see if HELPLIB is already OPENed. If it is, skip the
: 1798      1779 2 | OPENING code and get right to the HELP text retrieval.
: 1799      1780 2
: 1800      1781 2 IF NOT (.LIBRARY_OPEN)
: 1801      1782 2 THEN
: 1802      1783 2     BEGIN
: 1803      1784 2     IF NOT (STATUS = LBR$INI_CONTROL (HELP_LIBINDEX, HELP_FUNCTION, HELP_TYPE))
: 1804      1785 2     THEN
: 1805      1786 2       ERR_EXIT (DSKQS_HELP_INIT, .STATUS);
: 1806      1787 2
: 1807      1788 2     IF NOT (STATUS = LBR$OPEN (HELP_LIBINDEX, LIBRARY_NAME, 0, HELP_DEFNAME))
: 1808      1789 2     THEN
: 1809      1790 2       ERR_EXIT (DSKQS_HELP_OPEN, .STATUS);
: 1810      1791 2
: 1811      1792 2     LIBRARY_OPEN = 1;                                ! flag library open
: 1812      1793 2
: 1813      1794 2 END;
: 1814      1795 2
: 1815      1796 2
: 1816      1797 2 | Get and display the HELP text. LBR$GET_HELP will call LIB$PUT_OUTPUT
: 1817      1798 2 | to print the HELP text.
: 1818      1799 2
: 1819      1800 2 IF NOT (STATUS = LBR$GET_HELP (HELP_LIBINDEX, 0, 0, 0, KEY_VECTOR[0],
: 1820      1801 2                               KEY_VECTOR[2],
: 1821      1802 2                               KEY_VECTOR[4],
: 1822      1803 2                               KEY_VECTOR[6],
: 1823      1804 2                               KEY_VECTOR[8],
: 1824      1805 2                               KEY_VECTOR[10],
: 1825      1806 2                               KEY_VECTOR[12]))
: 1826      1807 2 THEN
: 1827      1808 2     ERR_EXIT (DSKQS_HELP_TEXT, .STATUS);
: 1828      1809 2
: 1829      1810 2     1 END;                                    ! end of routine ACT_HELP

```

.PSECT SPLIT\$,NOMRT,NODE\$E,2

42 4C 48 2E 3A 50 4C 45 48 24 53 59 53 0041C P.AAU:	.ASCII \SYSSHELP:.HLB\	:
0000000D 0042C P.AAT:	.BLKB 3	
00000000 00430	.LONG 13	
	.ADDRESS P.AAU	

41 54 4F 55 51 4B 53 49 44	00434 P.AAW:	.ASCII \DISKQUOTA\	:
	0043D	.BLKB 3	
00000009	00440 P.AAV:	.LONG 9	
00000000	00444	.ADDRESS P.AAW	
		.PSECT S0WNS,NOEXE,2	
00000001	0026C HELP_FUNCTION:	.LONG 1	
00000003	00270 HELP_TYPE:	.LONG 3	
	00274 HELP_LIBINDEX:	.BLKB 4	
	00278 LIBRARY_OPEN:	.BLKB 4	
		HELP_DEFNAME= P.AAT	
		LIBRARY_NAME= P.AAV	
		.EXTRN LBR\$INI CONTROL	
		.EXTRN LBR\$OPEN, LBR\$GET_HELP	
		.PSECT SCODE\$,NOWRT,2	
54 00000000G	00 001C 00000	.ENTRY ACT HELP. Save R2,R3,R4	1707
53 00000000	EF 9E 00002	MOVAB LIB\$STOP, R4	EX
4C 04	A3 E8 00010	MOVAB HELP LIBINDEX, R3	
	FC A3 9F 00014	BLBS LIBRARY OPEN, 38	1782
	F8 A3 9F 00017	PUSHAB HELP_TYPE	KE
		PUSHAB HELP_FUNCTION	1785
00000000G 00	03 FB 0001C	PUSHL R3	
52	50 D0 00023	CALLS #3. LBR\$INI_CONTROL	DI
08	52 E8 00026	MOVL R0, STATUS	
	52 DD 00029	BLBS STATUS, 18	
	8F DD 0002B	PUSHL STATUS	
64 004500F0	02 FB 00031	CALLS #2. LIB\$STOP	SG
00000000	EF 9F 00034	PUSHAB HELP_DEFNAME	
	7E D4 0003A	CLRL -(SP)	1789
00000000	FF 9F 0003C	PUSHAB LIBRARY_NAME	SO
00000000G 00	53 DD 00042	PUSHL R3	
52	04 FB 00044	CALLS #4. LBR\$OPEN	
08	50 D0 0004B	MOVL R0, STATUS	
	52 E8 0004E	BLBS STATUS, 28	
	52 DD 00051	PUSHL STATUS	
	8F DD 00053	PUSHL #4522232	
04 64 A3 004500F8	02 FB 00059	CALLS #2. LIB\$STOP	1791
	01 D0 0005C	MOVL #1. LIBRARY OPEN	-L
	C3 9F 00060	PUSHAB KEY_VECTOR+48	
	C3 9F 00064	PUSHAB KEY_VECTOR+40	1793
	C3 9F 00068	PUSHAB KEY_VECTOR+32	1806
	C3 9F 0006C	PUSHAB KEY_VECTOR+24	1805
	C3 9F 00070	PUSHAB KEY_VECTOR+16	1804
	C3 9F 00074	PUSHAB KEY_VECTOR+8	1803
	C3 9F 00078	PUSHAB KEY_VECTOR	1802
	7E 7C 0007C	CLRL -(SP)	1801
	7E D4 0007E	CLRL -(SP)	1800
	S3 DD 00080	PUSHL R3	

00000000G	00	08	FB 00082	CALLS #11, LBR\$GET_HELP
52		20	DD 00089	MOVL R0, STATUS
08		22	E8 0008C	BLBS STATUS, 48
	00450100	52	DD 0008F	PUSHL STATUS
		8F	DD 00091	PUSHL #4522240
64		C2	FB 00097	CALLS #2, LIB\$STOP
50		01	DD 0009A	MOVL #1, R0
		04	0009D	RET

; Routine Size: 158 bytes, Routine Base: SCODES + 0872

1831 1812 1 GLOBAL ROUTINE MAIN_HANDLER (SIGNAL_VEC, MECHANISM) =
1832 1813 1
1833 1814 1 !++
1834 1815 1
1835 1816 1 Functional Description:
1836 1817 1
1837 1818 1 This routine is the main condition handler for the DISKQUOTA utility.
1838 1819 1 It receives a signal which is either an internal error code or a
1839 1820 1 standard system status. If the former, the appropriate message is
1840 1821 1 formatted and printed. For the latter, the condition is simply
1841 1822 1 resignalled.
1842 1823 1
1843 1824 1 Calling Sequence:
1844 1825 1 standard
1845 1826 1
1846 1827 1 Input Parameters:
1847 1828 1 none
1848 1829 1
1849 1830 1 Implicit Inputs:
1850 1831 1 none
1851 1832 1
1852 1833 1 Output Parameters:
1853 1834 1 none
1854 1835 1
1855 1836 1 Implicit Outputs:
1856 1837 1 none
1857 1838 1
1858 1839 1 Routines Called:
1859 1840 1 none
1860 1841 1
1861 1842 1 Routine Value:
1862 1843 1 none
1863 1844 1
1864 1845 1 Signals:
1865 1846 1 none
1866 1847 1
1867 1848 1 Side Effects:
1868 1849 1 none
1869 1850 1
1870 1851 1 !--
1871 1852 1
1872 1853 2 BEGIN
1873 1854 2
1874 1855 2 MAP
1875 1856 2 SIGNAL_VEC : REF BBLOCK, ! signal vector arg
1876 1857 2 MECHANISM : REF BBLOCK; ! mechanism vector arg
1877 1858 2
1878 1859 2 LOCAL
1879 1860 2 FORMAT_DESC : VECTOR [2], ! string descriptor for message format
1880 1861 2 P : REF VECTOR [,BYTE], ! string pointer
1881 1862 2 ERR_CODE : BBLOCK [4]; ! error status code
1882 1863 2
1883 1864 2 EXTERNAL ROUTINE
1884 1865 2 LIB\$PUT_OUTPUT : ADDRESSING_MODE (GENERAL);
1885 1866 2
1886 1867 2
1887 1868 2 ! Get the signal code. If it is one of ours, get the message string and

```

1888 1869 2 ! do formatting as necessary.
1889 1870 2
1890 1871 2
1891 1872 2 ERR_CODE = SIGNAL_VEC[CHFSL_SIG_NAME];
1892 1873 2 IF .ERR_CODE[STSSV_FAC_NO] EQL FAC_CODE
1893 1874 2 THEN
1894 1875 2 BEGIN
1895 1876 3 ERR_CODE = .ERR_CODE[STSSV_MSG_NO];
1896 1877 3 P = .MESSAGE_TABLE[ERR_CODE];
1897 1878 3 FORMAT_DESC[0] = .P[1];
1898 1879 3 FORMAT_DESC[1] = .P + 2;
1899 1880 3 OUTPUT_DESC[0] = OUTPUT_LENGTH;
1900 1881 2
1901 P 1882 2 SFAOL (CTRSTR = FORMAT_DESC[0],
1902 P 1883 3 OUTLEN = OUTPUT_DESC[0],
1903 P 1884 3 OUTBUF = OUTPUT_DESC[0],
1904 P 1885 3 PRMLST = SIGNAL_VEC[CHFSL_SIG_ARG1]
1905 1886 3 );
1906 1887 3 LIB$PUT_OUTPUT (OUTPUT_DESC);
1907 1888 2
1908 1889 2 ! If there is a signal argument remaining, it is a system error status.
1909 1890 2 Convert its severity to error and signal it.
1910 1891 2
1911 1892 2
1912 1893 3 ERR_CODE = 0;
1913 1894 3 IF .SIGNAL_VEC[CHFSL_SIG_ARGS] GTRU .P[0] + 3
1914 1895 3 THEN
1915 1896 4 BEGIN
1916 1897 4 ERR_CODE = .VECTOR [SIGNAL_VEC[CHFSL_SIG_ARG1], .P[0]];
1917 1898 3 END;
1918 1899 2
1919 1900 2
1920 1901 2 IF .ERR_CODE NEQ 0
1921 1902 2 THEN
1922 1903 2 BEGIN
1923 1904 3 ERR_CODE[STSSV_SEVERITY] = STSSK_ERROR;
1924 1905 3 SIGNAL (.ERR_CODE);
1925 1906 2 END;
1926 1907 2
1927 1908 2 MECHANISM[CHFSL_MCH_SAVRO] = 1;
1928 1909 2 IF .BBLOCK [SIGNAL_VEC[CHFSL_SIG_NAME], STSSV_SEVERITY] EQL STSSK_SEVERE
1929 1910 2 THEN
1930 1911 2 BEGIN
1931 P 1912 3 SIOIW (CHAN = CHANNEL,
1932 1913 3 FUNC = IOS_DEACCESS);
1933 1914 2
1934 1915 2 SUNWIND (DEPADR = MECHANISM[CHFSL_MCH_DEPTH]);
1935 1916 2 END;
1936 1917 2
1937 1918 2 RETURN SS$_CONTINUE;
1938 1919 2
1939 1920 1 END;

```

.EXTRN SYSSFAOL

				007C 00000	.ENTRY	MAIN_HANDLER, Save R2,R3,R4,R5,R6	: 1812
				FF 9E 00002	MOVAB	OUTPUT_DESC, R6	
				08 C2 00009	SUBL2	#8, SP	
				54 04 00004	MOVL	SIGNAL_VEC, R4	
				52 04 00010	MOVL	4(R4), ERR_CODE	
00000045	8F	52	OC	10 ED 00014	CMPZV	#16, #12, ERR_CODE, #69	
				49 12 0001D	BNEQ	1\$	
				53 00000000'EF42	EXTZV	#3, #13, ERR_CODE, ERR_CODE	
				6E 01 A3 9A 0002C	MOVL	MESSAGE_TABLE[ERR_CODE], P	
				AE 02 A3 9E 00030	MOVZBL	1(P), FORMAT_DESC	
				66 84 8F 9A 00035	MOVAB	2(R3), FORMAT_DESC+4	
				55 08 A4 9E 00039	MOVZBL	#132, OUTPUT_DESC	
				55 DD 0003D	MOVAB	8(R4), R5	
				56 DD 0003F	PUSHL	R5	
				56 DD 00041	PUSHL	R6	
				00000000G 00 0C AE 9F 00043	PUSHAB	FORMAT_DESC	
				04 FB 00046	CALLS	#4, SYSSFAOL	
				56 DD 0004D	PUSHL	R6	
				00000000G 00 01 FB 0004F	CALLS	#1, LIB\$PUT_OUTPUT	
				52 D4 00056	CLRL	ERR_CODE	
				51 03 63 9A 00058	MOVZBL	(P), R0	
				51 A0 9E 0005B	MOVAB	3(R0), R1	
				51 64 D1 0005F	CMLP	(R4), R1	
				04 1B 00062	BLEQU	1\$	
				52 6540 DD 00064	MOVL	(R5)[R0], ERR_CODE	
				52 D5 00068 1\$: 0E 13 0006A	TSTL	ERR_CODE	
				02 F0 0006C	BEQL	2\$	
				52 DD 00071	INSV	#2, #0, #3, ERR_CODE	
				01 FB 00073	PUSHL	ERR_CODE	
				00 0C 0007A 08 AC DD 0007A 2\$: 01 DO 0007E	CALLS	#1, LIB\$SIGNAL	
				00 ED 00082	MOVL	MECHANISM, R3	
				25 12 00088	MOVL	#1, 12(R3)	
				7E 7C 0008A	CMPZV	#0, #3, 4(R4), #4	
				7E 7C 0008C	BNEQ	3\$	
				7E 7C 0008E	CLRQ	-(SP)	
				7E 7C 00090	CLR2	-(SP)	
				7E FEE4 34 7D 00092	CLRQ	-(SP)	
				7E C6 3C 00095	CLRQ	-(SP)	
				7E D4 0009A	MOVZWL	#52, -(SP)	
				0C FB 0009C	CHANNEL, -(SP)		
				7E D4 000A3	CLRL	-(SP)	
				00 000000G 00 08 A3 9F 000A5	CALLS	#12, SYSSQIOW	
				02 FB 000A8	CLRL	-(SP)	
				50 01 D0 000AF 3\$: 01 D0 000AF 04 000B2	PUSHAB	8(R3)	
				01 D0 000AF 04 000B2	CALLS	#2, SYSSUNWIND	
				01 D0 000AF 04 000B2	MOVL	#1, R0	
				01 D0 000AF 04 000B2	RET		

; Routine Size: 179 bytes, Routine Base: \$CODES + 0910

1941 1921 1 GLOBAL ROUTINE EXIT_HANDLER: NOVALUE =
1942 1922 1
1943 1923 1 :++
1944 1924 1
1945 1925 1 Functional Description:
1946 1926 1
1947 1927 1 This routine is called by the OS on exit (for whatever reason) from
1948 1928 1 the DISKQUOTA utility. This routine must ensure that DISKQUOTA did
1949 1929 1 not leave things in an awkward state.
1950 1930 1
1951 1931 1 Calling Sequence:
1952 1932 1 standard
1953 1933 1
1954 1934 1 Input Parameters:
1955 1935 1 none
1956 1936 1
1957 1937 1 Implicit Inputs:
1958 1938 1 none
1959 1939 1
1960 1940 1 Output Parameters:
1961 1941 1 none
1962 1942 1
1963 1943 1 Implicit Outputs:
1964 1944 1 none
1965 1945 1
1966 1946 1 Routines Called:
1967 1947 1 none
1968 1948 1
1969 1949 1 Routine Value:
1970 1950 1 none
1971 1951 1
1972 1952 1 Signals:
1973 1953 1 none
1974 1954 1
1975 1955 1 Side Effects:
1976 1956 1 none
1977 1957 1
1978 1958 1 --
1979 1959 1
1980 1960 2 BEGIN
1981 1961 2
1982 1962 2
1983 1963 2 Make sure that DISKQUOTA did not leave a volume LOCKED.
1984 1964 2
1985 1965 2
1986 1966 2 IF .CLEANUP_FLAGS[CLF_UNLOCK]
1987 1967 2 THEN
1988 1968 2 BEGIN
1989 1969 2 CHSFILL (0, FIBSC_LENGTH, QUOTA_FIB);
1990 1970 2 QUOTA_FIB[FIBSW_CTRLFUNC] = FIBSC_UNLK_VOL;
P 1971 1971 2 \$OJOW((CHAN = .CHANNEL,
1992 P 1972 1972 2 FUNC = IOS_ACPCONTROL,
1993 P 1973 1973 2 P1 = OFIB_DESC
1994 1974 2);
1995 1975 2 END;
1996 1976 2
1997 1977 1 END;

: end of routine EXIT_HANDLER

			007C 00000	.ENTRY EXIT HANDLER, Save R2,R3,R4,R5,R6	: 1921
0040	BF	00	EF 9E 00002	MOVAB CLEANUP_FLAGS, R6	:
			66 E9 00009	BLBC CLEANUP_FLAGS, 1\$: 1966
			00 2C 0000C	MOVCS #0, (SPT, #0, #64, QUOTA_FIB	: 1969
			A6 00013		
		5A A6	08 B0 00015	MOVW #8, QUOTA_FIB+22	: 1970
			7E 7C 00019	CLRQ -(SP)	: 1974
			7E 7C 0001B	CLRQ -(SP)	
			7E D4 0001D	CLRL -(SP)	
			C6 9F 0001F	PUSHAB QFIB_DESC	
			7E 7C 00023	CLRQ -(SPT)	
		7E FECB	38 7D 00025	MOVQ #56, -(SP)	
			C6 3C 00028	MOVZWL CHANNEL, -(SP)	
			7E D4 0002D	CLRL -(SP)	
		00000000G	00	CALLS #12, SYSSQIOW	
			0C FB 0002F		
			04 00036 1\$:	RET	: 1977

: Routine Size: 55 bytes. Routine Base: \$CODE\$ + 09C3

1999 1 GLOBAL ROUTINE COMMON_IO (EFN,CHAN,FUNC,IOSTS,ASTADR,ASTPRM,P1,P2,P3,P4,P5,P6)=
2000 1979 1
2001 1980 1 !++
2002 1981 1
2003 1982 1 FUNCTIONAL DESCRIPTION:
2004 1983 1
2005 1984 1 This routine simply executes a \$QIOW call with the parameters
2006 1985 1 supplied.
2007 1986 1
2008 1987 1 CALLING SEQUENCE:
2009 1988 1 COMMON_IO (EFN,CHAN,FUNC,IOSTS,ASTADR,ASTPRM,P1,P2,P3,P4,P5,P6)
2010 1989 1
2011 1990 1 INPUT PARAMETERS:
2012 1991 1 As to \$QIOW
2013 1992 1
2014 1993 1 IMPLICIT INPUTS:
2015 1994 1 NONE
2016 1995 1
2017 1996 1 OUTPUT PARAMETERS:
2018 1997 1 NONE
2019 1998 1
2020 1999 1 IMPLICIT OUTPUTS:
2021 2000 1 NONE
2022 2001 1
2023 2002 1 ROUTINE VALUE:
2024 2003 1 As to \$QIOW
2025 2004 1
2026 2005 1 SIDE EFFECTS:
2027 2006 1 As to \$QIOW
2028 2007 1
2029 2008 1 !--
2030 2009 1
2031 2010 2 BEGIN
2032 2011 2
2033 2012 2 BUILTIN
2034 2013 2 AP,
2035 2014 2 CALLG;
2036 2015 2
2037 2016 2 EXTERNAL ROUTINE
2038 2017 2 SYSSQIOW : ADDRESSING_MODE (GENERAL);
2039 2018 2
2040 2019 2
2041 2020 2 ! We simply pass the call and its parameters along to \$QIOW.
2042 2021 2
2043 2022 2
2044 2023 2 CALLG (.AP, SYSSQIOW)
2045 2024 2
2046 2025 1 END; ! End of routine COMMON_IO

00000000G 00

0000 00000
6C FA 00002
04 00009.ENTRY COMMON IO, Save nothing
CALLG (AP), SYS\$QIOW
RET: 1978
: 2023
: 2025

: Routine Size: 10 bytes, Routine Base: \$CODES + 09FA

: 2047 2026 1
: 2048 2027 1 END
: 2049 2028 0 ELUDOM

-S

PS
--
SP-L
-S

.EXTRN LIB\$SIGNAL, LIB\$STOP

PSECT SUMMARY

Name	Bytes	Attributes
\$MSG_TEXT	1804	NOVEC,NOWRT, RD ,NOEXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(0)
\$MSG_INDEX	132	NOVEC,NOWRT, RD ,NOEXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)
\$OWN\$	636	NOVEC, WRT, RD ,NOEXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)
SPLITS	1096	NOVEC,NOWRT, RD ,NOEXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)
-LIB\$KEY0\$	30	NOVEC,NOWRT, RD : EXE: SHR, LCL, REL, CON, PIC,ALIGN(1)
-LIB\$STATES	304	NOVEC,NOWRT, RD : EXE: SHR, LCL, REL, CON, PIC,ALIGN(1)
-LIB\$KEY1\$	110	NOVEC,NOWRT, RD : EXE, SHR, LCL, REL, CON, PIC,ALIGN(1)
\$CODES	2564	NOVEC,NOWRT, RD : EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)

Library Statistics

File	Symbols			Pages Mapped	Processing Time
	Total	Loaded	Percent		
\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	88	0	1000	00:01.9
\$255\$DUA28:[SYSLIB]TPAMAC.L32;1	42	30	71	14	00:00.2

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LISS:DISKQUOTA/OBJ=OBJ\$:DISKQUOTA MSRC\$:DISKQUOTA/UPDATE=(ENH\$:DISKQUOTA)

Size: 2564 code + 4112 data bytes
 Run Time: 01:26.1
 Elapsed Time: 02:55.3
 Lines/CPU Min: 1413
 Lexemes/CPU-Min: 57801
 Memory Used: 366 pages
 Compilation Complete

0104 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

DISMOUNT
MAP

DISKQUOTA
MAP

DISMOUNT
MAP

DISMNTSHR
MAP

DISKQ

DISKQUOTA
MAP

DISPLAY